



**ONLINE CLUSTER ANALYSIS SUPPORTING REAL TIME ANOMALY
DETECTION IN HYPERSPECTRAL IMAGERY**

GRADUATE RESEARCH PAPER

Elwood T. Waddell, Jr., Maj, USAF

AFIT-ENS-GRP-13-J-25

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC
RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the United States Government and is not subject to copyright protection in the United States.

**ONLINE CLUSTER ANALYSIS SUPPORTING REAL TIME ANOMALY
DETECTION IN HYPERSPECTRAL IMAGERY**

GRADUATE RESEARCH PAPER

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the
Degree of Master of Science in Operations Analysis

Elwood T. Waddell, Jr., BS, MS

Major, USAF

June 2013

DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC
RELEASE; DISTRIBUTION UNLIMITED

ONLINE CLUSTER ANALYSIS SUPPORTING REAL TIME ANOMALY
DETECTION IN HYPERSPECTRAL IMAGERY

Elwood T. Waddell, Jr., BS, MS
Major, USAF

Approved:

Dr. Kenneth W. Bauer

Date

Abstract

Ongoing work in anomaly detection in hyperspectral images has shown that cluster analysis performed in appropriate principal component subspaces can enhance the performance of detectors such as the Reed-Xiaoli detector and its derivatives. Numerous operational considerations motivate the development of an online or incremental clustering algorithm, which can perform clustering as pixels of the image are collected in real time rather than waiting until the full image is complete. Such an algorithm is developed by combining key elements of existing clustering algorithms from related domains. The parameters of the algorithm are tuned and performance of the algorithm is assessed using a set of actual hyperspectral images by exploiting key attributes of an appropriate principal component sub-space. A byproduct of the clustering algorithm is a rudimentary anomaly detector which demonstrates the feasibility of cluster based outlier detection in hyperspectral imagery.

To my Wife, our Daughter, and my Parents

Acknowledgments

I would first like to thank my research advisor, Dr. Bauer, for suggesting this topic and being willing to advise an Intermediate Developmental Education (IDE) student. His guidance and support have been more than instrumental in completing this project in a timely manner. The background material required to complete this project came primarily from one of his courses, and his talented instruction has been a joy to experience. I would also like to thank the IDE class advisor, Dr. Miller, for patiently fielding the many process and program related questions that invariably arose throughout the year.

The assistance from the staff of the Center of Operations Analysis has been critical to my completing this program, particularly with regard to computer resources. Mr. John Duke and Mr. Mark Fryman have been especially helpful and supportive. Similarly, Mr. Trevor Bihl in the Sensor Fusion lab has been incredibly helpful.

Other students who have worked in related areas include CPT James Jablonski, Capt Andrew Bigley, and Ms. Kelly Bush, all of whom gladly fielded many of my questions. A special thanks also goes to my fellow commiserators in the AFIT IDE program, particularly Maj John Isacco, Maj Adam Simoncic, and Maj Meghan Szwarc. Lastly, I cannot thank Maj Laura Waddell enough for her extraordinary technical editing and personal support.

Elwood T. Waddell, Jr.

Table of Contents

	Page
Abstract.....	iv
Acknowledgments.....	vi
Table of Contents	vii
List of Figures	x
List of Tables	xv
List of Equations	xvi
I. Introduction	1
General Issue.....	1
Research Objectives	2
Research Focus.....	2
Limitations	2
Implications.....	3
Preview.....	3
II. Literature Review	4
Chapter Overview	4
Hyperspectral Imagery	4
Principal Components Analysis (PCA).....	7
Anomaly Detection	12
Clustering	15
<i>K-means</i>	21
<i>ISODATA</i>	22
<i>X-means</i>	23
<i>DBSCAN</i>	24
Summary	25
III. Methodology	27
Chapter Overview	27
Image Data Set	27
Preliminary Experiments on the Distances Measures in the Principal Component	
Subspace.....	28
<i>Condition Number Test</i>	28
<i>Clustering Test</i>	29

<i>Eigen Values by Signature Test</i>	29
<i>Eigen Values by Combined Signatures Test</i>	30
<i>Loadings of Paired Signatures</i>	30
<i>Distance Plots</i>	31
Algorithm Description	31
<i>High Level Structure</i>	32
<i>Initialization</i>	34
<i>Cluster Membership Selection</i>	38
<i>Remove Old Data</i>	44
<i>Improve Structure (Cluster Merging)</i>	44
<i>Improve Structure (Cluster Splitting)</i>	50
<i>Anomaly Detection</i>	52
Algorithm Assessment Methodology.....	54
<i>Algorithm Verification and Fault Detection Considerations</i>	54
<i>Tuning</i>	55
<i>Performance Validation</i>	61
Summary	61
IV. Analysis and Results.....	62
Chapter Overview	62
Preliminary Confirmation of Insufficiency of Euclidean Distance for Clustering	62
<i>Condition Number Test Results</i>	63
<i>Clustering Test</i>	65
Characterization of the Principal Component Subspace and Mahalanobis Distance	
Behavior	70
<i>Eigen Values by Signature Test Results</i>	70
<i>Eigen Values by Combined Signature Test Results</i>	72
<i>Loadings of Paired Signatures Results</i>	73
<i>Distance Plots Results</i>	75
Tuning Procedures on a Single Image	87
<i>Split Cluster Behavior</i>	96
Sensitive to Principal Component Selection	98
Summary	99
V. Conclusions and Recommendations	100
Chapter Overview	100
Conclusions of Research	100
Significance of Research.....	101
Recommendations for Future Research	101
Summary	104
Appendix A.....	105
Appendix B	106

Appendix C	107
Appendix D	126
Appendix E	145
Appendix F.....	146
Appendix G.....	147
Bibliography	148
Vita.....	154

List of Figures

	Page
Figure 1: A Hyperspectral Image.....	5
Figure 2: Two Dimensional Example of PCA Translation and Rotation	8
Figure 3: Looney's High Level Algorithm	18
Figure 4: Online Clustering Algorithm Flow Chart.....	33
Figure 5: Spectra of Hyperspectral Image with Discarded Bands Exploded.....	35
Figure 6: Cluster Membership Determination (Mahalanobis Distance Motivation)	39
Figure 7: Cluster Membership Determination (Euclidean Rules Two through Four)	42
Figure 8: Potential Error Addressed by Combining Existing Clusters	45
Figure 9: Potential Error Addressed by Splitting an Existing Cluster	51
Figure 10: Detector Behavior.....	53
Figure 11: Area Under the ROC Curve For Single Points	59
Figure 12: Maximum K-means Covariance Condition Numbers (\log_{10} scale)	64
Figure 13: Median K-means Covariance Condition Numbers (\log_{10} scale).....	64
Figure 14: Minimum K-means Covariance Condition Numbers (\log_{10} scale).....	65
Figure 15: Randomly Selected HSI Image	66
Figure 16: Hyperspectral Image Pixels Plotted in the First Two Principal Components	67
Figure 17: Clustering in 2 PC's using K-means (k=3) & Euclidean Distances.....	68
Figure 18: Clustering in 2 PC's using K-means (k=5) & Euclidean Distances.....	69
Figure 19: Spectrum of Various Pixel Types.....	70
Figure 20: Eigen values of Selected Pixel Covariance Matrices	72
Figure 21: Eigen values of Selected Combined Pixel Covariance Matrices	73

Figure 22: 1 st Principal Component Loading Plot	74
Figure 23: Mean (by band) divided by standard deviation	75
Figure 24: Mahalanobis Distance of Selected Pixels to Mean Road Value	76
Figure 25: Mahalanobis Distance of Selected Pixels to Mean Tree Value.....	76
Figure 26: Mahalanobis Distance of Selected Pixels to Mean Road Value (with Anomalies).....	77
Figure 27: Mahalanobis Distance of Selected Pixels to Mean Tree Value (with Anomalies).....	78
Figure 28: Euclidean Distance of Selected Pixels to Mean Tree Value	79
Figure 29: Selected Pixels plotted in the first two principal components.....	80
Figure 30: Principal Component Spectrum of Various Pixel Types.....	81
Figure 31: Principal Component Spectrum of Various Pixel Types (1-20).....	82
Figure 32: Principal Component Spectrum of Various Pixel Types (2-4).....	83
Figure 33: Principal Component Plot (1 st verses 3 rd).....	83
Figure 34: Principal Component Spectrum of Various Pixel Types (14-20).....	84
Figure 35: Pixels Plotted in Natural Color in Selected Principal Components	85
Figure 36: Potential Analysis Tool GUI Example	86
Figure 37: Total Number of Clusters (log ₁₀ scale)	88
Figure 38: Distribution of Clusters (log ₁₀ scale)	89
Figure 39: Time required to execute the algorithm in log ₁₀ (seconds).....	90
Figure 40: Number of Mismatches between in Mahalanobis and Euclidean Distances..	91
Figure 41: AUC and Execution Time Comparison.....	92
Figure 42: AUC and Execution Time Comparison (Focused Region)	93

Figure 43: ROC Summary Plot (Table 5 Settings)	94
Figure 44: ROC Summary Plot (Table 6 Settings)	96
Figure 45: Image Clustering (Table 7 Settings).....	98
Figure 46: Clustering of Image 6 with Sand Generated Principal Components (Table 7 Settings)	99
Figure 47: Summary Results for Lambda =1, 10, 15, 20, 25, 30, & 35	105
Figure 48: Detector Performance for Lambda=1	106
Figure 49: Summary of Clustering Performance (Table 5 Settings)	107
Figure 50: Image 1 Results (Table 5 Settings).....	108
Figure 51: Image 2 Results (Table 5 Settings).....	109
Figure 52: Image 3 Results (Table 5 Settings).....	110
Figure 53: Image 4 Results (Table 5 Settings).....	111
Figure 54: Image 5 Results (Table 5 Settings).....	112
Figure 55: Image 6 Results (Table 5 Settings).....	113
Figure 56: Image 7 Results (Table 5 Settings).....	114
Figure 57: Image 8 Results (Table 5 Settings).....	115
Figure 58: Image 9 Results (Table 5 Settings).....	116
Figure 59: Image 10 Results (Table 5 Settings).....	117
Figure 60: Image 11 Results (Table 5 Settings).....	118
Figure 61: Image 12 Results (Table 5 Settings).....	119
Figure 62: Image 13 Results (Table 5 Settings).....	120
Figure 63: Image 14 Results (Table 5 Settings).....	121
Figure 64: Image 15 Results (Table 5 Settings).....	122

Figure 65: Image 16 Results (Table 5 Settings).....	123
Figure 66: Image 17 Results (Table 5 Settings).....	124
Figure 67: Image 18 Results (Table 5 Settings).....	125
Figure 68: Summary of Clustering Performance (Table 6 Settings)	126
Figure 69: Image 1 Results (Table 6 Settings).....	127
Figure 70: Image 2 Results (Table 6 Settings).....	128
Figure 71: Image 3 Results (Table 6 Settings).....	129
Figure 72: Image 4 Results (Table 6 Settings).....	130
Figure 73: Image 5 Results (Table 6 Settings).....	131
Figure 74: Image 6 Results (Table 6 Settings).....	132
Figure 75: Image 7 Results (Table 6 Settings).....	133
Figure 76: Image 8 Results (Table 6 Settings).....	134
Figure 77: Image 9 Results (Table 6 Settings).....	135
Figure 78: Image 10 Results (Table 6 Settings).....	136
Figure 79: Image 11 Results (Table 6 Settings).....	137
Figure 80: Image 12 Results (Table 6 Settings).....	138
Figure 81: Image 13 Results (Table 6 Settings).....	139
Figure 82: Image 14 Results (Table 6 Settings).....	140
Figure 83: Image 15 Results (Table 6 Settings).....	141
Figure 84: Image 16 Results (Table 6Table 6Table 5 Settings)	142
Figure 85: Image 17 Results (Table 6 Settings).....	143
Figure 86: Image 18 Results (Table 6 Settings).....	144
Figure 87: Image 6 Results with Split Cluster Enabled (Table 7 Settings)	145

Figure 88: Image 6 Results with Sand Generated Principal Components (Table 6

Settings) 146

List of Tables

	Page
Table 1: Elements of a Cluster.....	36
Table 2: Experimental Design Factors and Levels	56
Table 3: Detector Outcome.....	57
Table 4: Response Variables.....	60
Table 5: Single Image Detector Optimal Selected Operating Parameters	93
Table 6: Preliminary Experimentation Suggested Operating Parameters.....	95
Table 7: Settings for Runs with Split Cluster Enabled	97

List of Equations

	Page
Equation 1	13
Equation 2	23
Equation 3	43
Equation 4	46
Equation 5	47
Equation 6	47
Equation 7	48
Equation 8	48
Equation 9	57
Equation 10	57
Equation 11	59

ONLINE CLUSTER ANALYSIS SUPPORTING REAL TIME ANOMALY DETECTION IN HYPERSPECTRAL IMAGERY

I. Introduction

General Issue

Recent advances in imaging have moved beyond multispectral to hyperspectral imaging. Hyperspectral images contain a host of available information, but require advanced techniques for image processing and anomaly detection. Hyperspectral imagery is especially well suited to finding unusual signatures in a background. This is particularly true of manmade objects in a natural setting, which motivates an interest in the technology from the defense community. For various reasons, the processing of a hyperspectral image as it is being collected or in real-time is of interest. Common anomaly detection methods, which have been adapted to detect anomalies as the image is being collected, can be enhanced by the use of cluster analysis to group similar signatures, particularly backgrounds, in the image into distinct groups or clusters. In order for a detector to benefit from a clustering of pixels in the image as the image is being processed, the clustering would need to occur as the image is being collected. Most common clustering algorithms operate on a set of data that is static, and complete as the time of cluster. Although several clustering algorithms have been developed to perform clustering as data enters the algorithm, no such algorithm has been applied to hyperspectral image processing and anomaly detection in real time.

Research Objectives

The first objective of this project is to demonstrate a relatively fast clustering algorithm that can cluster hyperspectral imagery pixels as an image is being collected. This differs from the previously demonstrated method of using a clustering method on an entire image prior to performing anomaly detection.

Research Focus

The implementation or development of a clustering algorithm that can cluster hyperspectral imagery in real time requires an understanding of the data to be clustered. Hyperspectral imagery often undergoes dimensionality reduction by projecting the collected data into a principal component subspace. This project must expend some effort in understanding the behavior of the hyperspectral data in the principal component subspace to both select or develop an appropriate algorithm and to select appropriate parameters, as required. Additionally, as the clustering must support anomaly detection, a demonstration of a detector which takes as input the clustered data, would be relevant. Appropriate detectors will be investigated for this reason.

Limitations

Only a few hyperspectral data sets are available for use. The applicability of the specific parameters that may be found through this project may only be relevant to the imaging systems used to determine the parameters; however, the method used to determine the parameters should be relevant to any given hyperspectral imaging system.

Implications

An online clustering algorithm for use with hyperpsetral imagery would not only improve anomaly detection performance, but also potentially allow for image segmentation or improved inputs for signature matching purposes. Such an algorithm may have utility in other domains which exhibit any of the exploitable features of the hyperspectral domain used to construct the algorithm.

Preview

Chapter 2 will cover the necessary background in hyperspectral imagery, principal component analysis, anomaly detection, and clustering to fully develop reasoning behind the algorithm developed chapter 3. Chapter 3 opens with a description of some simple tests that can indicate *a priori* whether or not the intended algorithm described might work in a given domain. The algorithm itself is then described, followed by the methods required to verify and validate the functionality of the algorithm. Chapter 4 presents the results from both the initial tests described in chapter 3 and the verification and validation testing described at the end of Chapter 3. Several interesting features of the principal component subspace will be described, suggesting a number of areas for further research. Fast online clustering will be demonstrated, as will anomaly detection as a byproduct of online clustering.

II. Literature Review

Chapter Overview

This chapter follows the pattern of (Bush, 2012) and (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007) in providing a background in hyperspectral imagery (HSI) before discussing principal component analysis (PCA) as a dimensionality reduction tool for HSI. Relevant, current work in anomaly detection in HSI is then presented to motivate the discussion on clustering and the need for an online clustering algorithm for HSI applications.

Hyperspectral Imagery

Hyperspectral imagery is similar to consumer digital photography, differing primarily in the range and resolution of frequencies recorded for each pixel in the image. Commercially available black and white digital cameras receive electromagnetic radiation in wavelengths corresponding to the human visible range, and project it onto a focal plane array which then converts the energy levels into an intensity, which is then recorded in the image. In commercially available color cameras, the wavelengths corresponding to the colors red, green, and blue are recorded at the focal plane on a pixel by pixel basis. Hyperspectral imaging systems collect a wider range of wavelengths and project them through more complex means onto a suite of sensors which record the energy levels in each continuous range of wavelengths, called a band, in the range, often at a finer resolution than the frequency separation between red, green, and blue wavelengths. While a digital camera will have 3 numerical values to record for a single pixel, a hyperspectral imaging system may have hundreds of values to record, a majority

of which will be outside the human visual range. Figure 1 shows a graphical representation of a hyperspectral image. The image was collected using the system described in (Rickard, Basedow, Zalewski, Silverglate, & Silverglate, 1993). The top layer is the image as would be seen by the human eye. The subsequent three layers show the intensity of each pixel in the red, green, and blue ranges, while all other layers, shown in gray as they cannot be seen by humans, represent the other bands which contain the vast majority of the information recorded by the system. The full spectrum recorded by the system of several representative pixels is also shown.

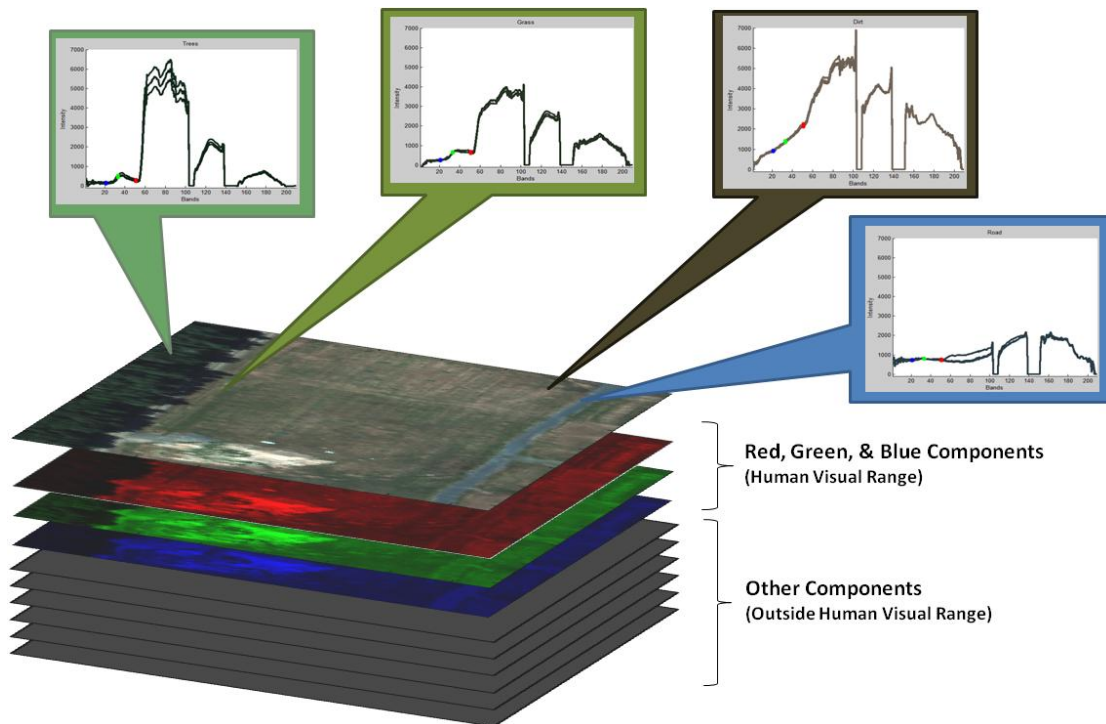


Figure 1: A Hyperspectral Image

The four spectral plots shown in Figure 1 each show three example pixels from parts of the image that contain four very different objects. Specifically, notional trees, grass, dirt and road pixel spectra are shown. Across the spectrum imaged by the sensor,

the intensity values in each band for each type of material are similar, while the intensities in each band between the objects clearly differ in some regions. The intensity of the pixels in the human visual range in this image accounts for only about 10% of the range of wavelengths collected by the sensor. The fact that different objects have different spectral signatures outside the human visual range means that two objects of the same “color” as perceived by a human eye can be easily distinguished from each other in a hyperspectral image. In military application, this could provide a method to defeat some types of camouflage, or match specific signatures for the detection or identification of objects or materials.

For computer vision and image processing applications including image segmentation (breaking the image into regions containing similar pixels and presumably, similar objects), object detection (finding something in an image), and object classification (determining the type of object shown in an image or subset of an image), HSI data provides a much richer source of information than monochromatic or grayscale images. Traditional image processing and computer vision techniques have focused on grayscale images due to their availability and their relative simplicity. Even recent introductory texts focus on the grayscale case, where the spatial relationships of the pixels contain a majority of the information useful in processing (Trucco & Verri, 1998).

Due to the increased dimensionality, HSI is fundamentally more difficult to process than traditional gray scale or three color images. As an example, the fundamental task of edge detection in grayscale images, which has traditionally relied on the spatial relationships between the pixels in the image, is considered well understood, with the Canny edge detector having seen popular use for over 20 years (Canny, 1986).

Meanwhile, edge detection in color images (a three dimensional rather than one dimensional case) is not considered thoroughly researched (Koschan & Abidi, 2005). This is the case despite the consideration of edge detection in color images as early as the late 1970's (Robinson, 1977). Recent work in this area for HSI included the use of cluster analysis of gradients in the image, but the resulting algorithm was computationally intensive (Dinh, Leitner, Paclik, & Duin, 2009). No clear "standard" algorithm for edge detection has appeared for HSI imagery.

While the development of methods for use in HSI processing is still fairly recent, the trend toward using the spectral more than spatial information for processing applications in the images appears well established for numerous applications. Some sources have sought to combine both spatial and spectral information including (Goovaerts, Jacquez, & Marcus, 2005) and (Messer, 2011), but in each case, spectral information (as reduced via some form of component analysis) is the basis of the algorithm. Also, the need for dimensionality reduction techniques to reduce hyperspectral images to a manageable size for processing in a reasonable space is widely accepted, with multiple techniques thoroughly researched (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007).

Principal Components Analysis (PCA)

PCA is used routinely used for dimensionality reduction in hyperspectral image processing. Simply put, this technique transforms the intensity information in each band into a different space by rotating the coordinate axes so that the variability in the first principal component contains the maximum amount of variance in the data set. The

second principal component, constructed to be orthogonal to the first, contains the maximum variance remaining in the data set subject to the orthogonality constraint. Each additional principal component is constructed likewise. The actual rotation matrix is provided by the Eigen decomposition of the covariance or correlation matrix. Eigen values provide a measure of associated variance captured in the axes defined by the associated Eigen vectors. The use of covariance or correlation produces different results, and each is appropriate for specific domains. The use of covariance based PCA is assumed for the remainder of this project. A thorough description of the technique is available in (Dillon & Goldstein, 1984). An example with data in two dimensions is shown in Figure 2.

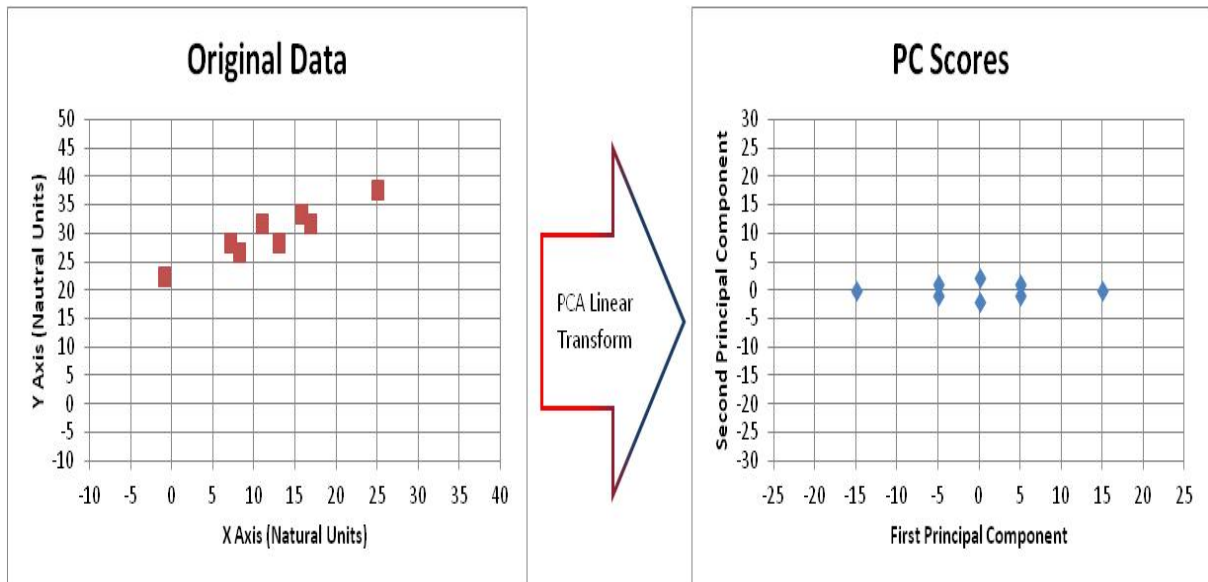


Figure 2: Two Dimensional Example of PCA Translation and Rotation

The red points represent a data set collected in natural units (on both the X and Y axes). The blue points are the PCA scores in two dimensions. In both set of axes, the scale is the same to illustrate the linear nature of the PCA transform. By convention, and

for convenience, the blue points are centered, with each axis having a mean of zero. This involved a translation of “down” 30 and “left” 12 from the mean of the original data. The data are also rotated clockwise by 30 degrees. For the blue points, the first principal component axis has the most variation. The variance of the first principal component values is roughly 78.6. Meanwhile, the second principal component has a variance of roughly 1.7. The two axes are completely uncorrelated, and a majority of the information relating to the Euclidean distance between the points can be expressed in only the first principal component.

The blue points, also referred to as the principal components scores, can be used in place of the original data for many forms of analysis. Unfortunately, this complicates the interpretation of the results of the analysis. In the example shown, a score in the first principal component is determined by a linear combination of both the original X and Y values, but is not exactly either one. In this simple example, the first principal component appears most closely related to the X value in the data. This could be expressed mathematically as a correlation between the first principal component scores and the actual x values in natural units. This correlation, taken for all scores and all variables, is referred to as the loadings matrix. If a given component is heavily loaded against a given original variable (or set of variables), those values drive the principal component value (more than the other variables). Viewing the loadings can greatly assist in the interpretation of the principal components.

In an ideal application, the first few principal components contain a majority of the total variation seen in the data, and retain the specific variation required to perform the desired processing task, allowing the other components to be discarded and reducing

the dimensionality required for task performance substantially. Essentially, PCA is used in this manner to create a subspace into which the data set is projected. While this simplifies processing for most applications, there is no guarantee that the largest (or several largest), principal components retain the needed information to perform a desired action. For non-linear behaviors, it is even possible that no single PCA will capture the appropriate information.

Assuming the use of PCA is appropriate, and the information required for a task is contained in the first several principal components, the decision as to which components should be retained for processing is non-trivial. Commonly discussed methods include those due to Kaiser, Cattell, and Horn. While introduced in 1965, the Horn's method was only recently implemented for a variety of applicable parameters by (Bigley, 2013).

For HSI applications, numerous methods have been considered for selecting the appropriate number of principal components for reduced dimensionality processing. Sources such as (Bush, 2012) and (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007) have selected the number of principal components to retain based on a few sample images used for tuning or training their algorithms. For the AutoGAD algorithm, which processes HSI images for anomaly detection, the maximum distance secant line (MDSL) was introduced to automatically select the number of PC's with substantial practical success (Johnson, 2008). Work in selecting non-consecutive PC's based on specific indices has been done by (Bihl, Bauer, & Friend, 2013).

A standard practice when using PCA for HSI processing is to calculate the PCA scores of an entire image (each point is the spectral response of a pixel, without regard to

spatial position in the image) before beginning further processing on the image. This is problematic for any image processing algorithm that seeks to begin processing an image prior to its complete collection. This issue was addressed by (Bush, 2012), by taking the PCA of the first several rows of a given image and assuming stable principal components. Recently, other methods of updating the principal components as data enters the system have become available.

Recursive PCA (RPCA) was described by (Li, Yue, Valle-Cervantes, & Qin, 2000). RPCA sought to address the issues associated with using a static PC set when the process varies with time. A recursive update to the covariance matrix is central to the method, and it is performed after each data point or batch of data points is collected. More recently, development of online PCA (OLPCA), suggests that principal components can be successfully updated over time based on key indicators rather than at each step (Tang, Yu, Chai, & Zhao, 2012). The key element to the efficiency of the OLPCA algorithm is the use of the residual (in contrast to the retained) principal component information to determine when the retained principal components have become insufficient to adequately capture the required amount of variability (for whatever purpose) in the developing process. When a pre-determined threshold of is exceeded, updates are made directly to the applicable covariance matrices central to the PCA methods, without reference to previously collected data. OLPCA is described as more accurate and faster than both moving window PCA (MWPCA) and recursive PCA (RPCA) for changing processes (Tang, Yu, Chai, & Zhao, 2012). The concept of using continuously collected data to update covariance matrices have also extended into

Adaptive Kernel PCA (or AKPCA) (Ding, Tian, & Xu, 2010) and moving window kernel PCA (Liu, Kruger, Littler, Xie, & Wang, 2009).

To this point in the discussion of PCA, it has been assumed that a single set of principal components is appropriate to capture the information in a single hyperspectral image. Use of PCA in this manner could be considered global rather than local dimensionality reduction. Were the correlation of various subsets of the image correlated differently, a method to perform PCA locally might be preferable. Such a method and motivating discussion is provided by (Chakrabarti & Mehrotra, 2000). Their method allows for multi-dimensional indexing using a specific tree structure, which greatly enhances computational efficiency, and allows precise and accurate reconstruction from the reduced dimensions even when the various sub-sets of the data differ greatly in their correlation. The determination of the local areas for dimensionality is, in fact, a clustering algorithm. For comparison, this algorithm was considered by (Kriegel, Kroeger, & Zimek, 2009) to be a slight variant of another clustering algorithm named ORCLUS, with a faster variant presented by (Li, Huang, Selke, & Yong, 2007).

Anomaly Detection

Anomaly detection can be summarized as finding things (items, areas, pixels, behaviors, signals, etc.) that are dissimilar to others which form the majority under consideration. While, a field of illicit crops in a forest of trees, an incongruent mineral deposit in a field or an algae bloom in an open ocean may each be an anomaly of interest to of civil law enforcement, commercial prospectors, or oceanographers, defense applications are generally more focused on finding artificial or manmade objects in

natural settings. Examples might include artillery emplacements, camouflaged artillery emplacements, buildings, tents, or vehicles. Hyperspectral imagery is uniquely suited to finding anomalies, particular if they have been disguised in terms of one part of the electromagnetic spectrum (such as the visual range), but not in terms of another area (such as near IR) where the object may still be distinct from the background.

Numerous detectors have been described for use in HSI applications. Surveys and comparisons are available from such sources as (Matteoli, Diani, & Corsini, 2010) and (Smetek & Bauer, A Comparison of Multivariate Outlier Detection Methods for Finding Hyperspectral Anomalies, 2008). The Reed-Xiaoli or RX detector first described by (Reed & Yu, 1990) is the basis of the most relevant family of detectors for this project. The detector primarily functions by comparing a single pixel to those inside a predetermined window size around it. The comparison is performed using a specific statistical test on the RX score which (Bush, 2012) notes is asymptotically equivalent to the Mahalanobis or statistical distance defined in Equation 1.

$$D_{MH} = \sqrt{(x_1 - x_2)S^{-1}(x_1 - x_2)^t} \quad \text{Equation 1} \quad (1)$$

Where:

x_1 = n dimensional vector for the candidate point

x_2 = n dimensional vector for the centroid of the window

S^{-1} = Inverse sample covariance matrix

t indicates transposition

The Mahalanobis distance is the multivariate equivalent of a z-score, where the inverse covariance matrix acts as the standard deviation in the traditional z-score calculation. Unlike standardizing each dimension independently, the use of the inverse covariance matrix accounts for correlation between the bands. For an application such as

the RX score, the applicable covariance matrix and mean vector are calculated from the pixels in the entire window (routinely, from the PCA scores of the pixels in the window after a dimensionality reduction step). An ideal case for detection would involve a single anomalous pixel in the window, as compared against the other non-anomalous pixels, but if an imaging system is designed to have sufficient resolution to capture a single object in multiple pixels, as might be expected if consideration were given to Johnson's criteria as shown in (Donohue, 1991), several anomalous pixels will be part of the background in the window, and will potentially affect the construction of the covariance matrix. The impact of having even a small number of outliers on a covariance matrix in HSI applications can be severe as demonstrated by (Smetek & Bauer, Finding Hyperspectral Anomalies Using Multivariate Outlier Detection, 2007). To mitigate the impacts of having anomalous pixels in the background, the locally adaptive iterative RX (IRX) detection algorithm was developed by (Taitano & Bauer, 2010). The algorithm systematically removes the declared anomalous pixels from the background calculations and repeats the detection on the image until no changes occur when anomalies are removed from the mean and covariance calculations. An additional innovation to address the spatial correlation was developed by (Williams, Towards the Mitigation of Correlation Effects in the Analysis of Hyperspectral Imagery with Extensions to Robust Parameter Design, 2012). In the Linear RX (LRX) and Iterative Linear RX (ILRX) detectors, the window used for the base RX and IRX algorithm is modified to be a row or set of rows in the image around the pixel to be assessed, rather than the traditional square or window. This increases the distance between pixels in the window, to attempt to decrease the effects of spatial correlation. The LRX and ILRX detectors have the

desirable property of being able to detect anomalies as an image is collected, assuming the image is being collected line by line. Line by line collection is standard for a push broom sensor, which is a common implementation for a hyperspectral imaging system.

While a single pixel could produce issues in covariance and mean calculations, the presence of several different backgrounds in a single window could also produce inflation of the covariance and adversely impact detection. One approach to mitigating the presence of several potential backgrounds in a single window is the use of clustering, to segment the background the image, as suggested by (Woodruff & Reiners, 2004) and (Hardin & Rocke, 2004). The use of clustering to separate the various backgrounds to improve the performance of the RX anomaly detector and change detection applications was demonstrated by (Carlotto, 2005). The idea was further explored, evaluated, and found useful by (Smetek, Hyperspectral Imagery Target Detection Using Improved Anomaly Detection and Signature Matching Methods, 2007).

Although clustering to improve the RX detector has been demonstrated, clustering to support the ILRX or RX detector as an image is being collected has not. A clustering method that clusters the pixels as the image is collected would be required to perform this task.

Clustering

Clustering is form of unsupervised learning, which seeks to place similar items into clusters. Clustering can be summarized as placing things (items, areas, pixels, behaviors, signals, etc.) into groups where the things in any given group are more similar to each other than they are to any of the things in the other groups. In this simple

construct, the groups are called clusters. While the concept of clustering is rather simply stated, the problem of grouping items together to minimize some similarity measure is NP-hard usually resulting in the use of a heuristic algorithm to perform clustering for practical applications (Jain A. K., Data clustering: 50 years beyond K-means, 2010).

Several texts provide background on common clustering algorithms and constructs including (Dillon & Goldstein, 1984) and (Duda, Hart, & Stork, 2001). Numerous extensive surveys on the field of clustering have also been performed. The survey by (Xu & Wunsch, 2005) provides a useful single summary of computational complexity of various popular clustering algorithms with an assessment of their ability to cluster high dimensional data sets and an evaluation of their performance on various benchmark data sets. A review by (Jain, Murty, & Flynn, 1999) provides a more focused “statistical pattern recognition perspective” of clustering algorithms. Another recent survey (Jain A. K., Data clustering: 50 years beyond K-means, 2010) indicates that literally thousands of clustering algorithms have been developed over the past 50 years for use in various domains. All of the surveys and texts cited here begin with the same basic taxonomy of clustering algorithms, splitting them into two major groups. One group, described nearly universally as partitional cluster, is generally concerned with optimizing some objective measure, and frequently involves optimization and iterative processing to seek the optimum. The other general class of methods, nearly universally known as hierarchical, seeks to place every object somewhere in a hierarchy of clusters. In the extreme, this places every object in a cluster by itself at the bottom of the hierarchy, and all objects together in a single cluster at the top of the hierarchy. These categories are further broken down and augmented differently depending on the source.

With such an abundance of algorithms and types of algorithms available, selecting the appropriate one for a given purpose is not at all straight forward. While it is possible to describe a set of axioms which a good clustering algorithm should meet, the impossibility theorem due to Kleinberg shows that it is impossible for any algorithm to behave in accordance with all these axioms (Jain A. K., Data clustering: 50 years beyond K-means, 2010). Attempting to evaluate all available clustering algorithms for a given purpose is certainly practically impossible. Fortunately, a clustering of the performance of a wide variety of clustering algorithms was performed by (Jain, Topchy, Law, & Buhmann, 2004). A major result of their paper was showing that clustering algorithms can be grouped by their behavior, and any member of a given cluster will likely have the same general behavior on a given type of data. The ramification being that only a few algorithms (one from each of a few various classes of algorithms) need be assessed to determine which ones might work best for clustering in a given domain. This fact, when combined with the results from (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007), indicates that distance based partitional clustering methods are more promising in the HSI domain than are hierarchical methods, allowing a focus on algorithms such as K-means, ISODATA, and similar derivatives.

A high level algorithm due to (Looney, 1997) shown in Figure 3, describes the general approach of numerous partitional algorithms. Similarity could be measured in any number of ways. Distances such as Euclidean, Mahalanobis, divergences, and correlation, are common, and appear to be popular in HSI processing. For some specific algorithms, such as K-means, this general algorithm is repeated until a single pass through the exemplars (or data elements) results in no change to cluster assignment,

which achieves a local optimum. A single pass through the algorithm may not guarantee any particular result or behavior (from an optimal perspective).

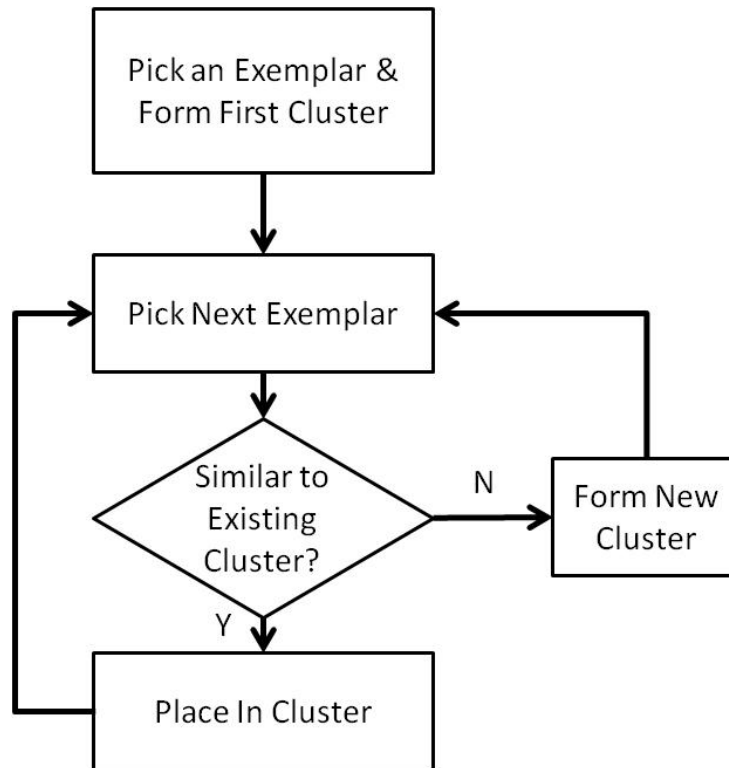


Figure 3: Looney's High Level Algorithm

Another common term that has become solidified in the vocabulary of clustering is the meaning of sub-space clustering. It is described succinctly by (Jain A. K., Data clustering: 50 years beyond K-means, 2010) as “finding clusters embedded in low-dimensional subspaces of the given high-dimensional data.” Clustering a data set using a subset of the principal component scores, as is common for HSI applications, is by definition subspace clustering, as mentioned in (Kriegel, Kroeger, & Zimek, 2009). Use of PCA scores rather than actual data values does not change the structure of the algorithm, such as the one shown in Figure 3. The algorithm simply sees the exemplars

(or data elements) in their subspace representation. From this perspective, nearly any clustering algorithm could be a subspace clustering algorithm, if the algorithm is presented with a data set that has been projected into some subspace.

Before beginning a discussion of specific relevant algorithms, it is important to mention that the development of clustering algorithms in various fields appears to have given rise to disconnects in nomenclature. Specifically, no single adjective is used to describe clustering algorithms that perform (or that could perform) clustering as data is collected. Further, the adjectives that are used do not have consistent meanings. The term “single pass” or “one pass” has seen somewhat consistent use to describe clustering that makes only one pass through the data to perform clustering. The term is used by (Cobb, 1988) to describe an algorithm he was evaluating. The algorithm was used to process multispectral image data that had been completely collected, but the nature of the algorithm inherently supports processing as data is collected. The term was also used by (O'Callaghan, Mishra, Meyerson, Guha, & Motwani, 2002) to describe a requirement of their “streaming-data” clustering algorithms. The phrase “streaming-data” was also used by (Aggarwal, Han, Wang, & Yu, 2003) while simultaneously invoking the term “evolving” to describe changing data or clustering. Their framework introduced a two component clustering method involving an “online component which periodically stores detailed summary statistics and an offline component which uses only this [*sic*] summary statistics.” The term “evolutionary clustering” is used by (Chakrabarti, Kumar, & Tomkins, 2006) to describe “processing time stamped data to produce a sequence of clusterings” and also invokes “online” to describe the general characteristic of processing data as it is collected. Their work also introduces the idea of minimizing some history

cost, or changes to cluster memberships assignments as data points enter the clustering algorithm. The terms “adaptive” and “online” are used to describe the use of the clustering algorithm due to (Wang, Masseglia, Guyet, Quiniou, & Cordier, 2009) for world wide web data processing, while the term “streaming” is taken to form the name of the algorithm (Str-DBSCAN). In contrast to other uses, “adaptive” was used by (Mok, Huang, & Kwok, 2012) to mean a clustering algorithm that works with a variety of other clustering algorithms, with no indication of clustering as more data is collected and “online” is used in the k-means algorithm implementation in (Mathworks, 2012) to describe a specific step of the algorithm, again without regard to any concept of clustering as data is collected. While “streaming” has been used to describe an update to the DBSCAN, the term “incremental” has also been used by (Chakraborty & Nagwani, Analysis and Study of Incremental DBSCAN Clustering Algorithm, 2011) to describe a modification to DBSCAN allowing it to cluster as data enters the algorithm. Similar work was done with the incremental k-means algorithm (Chakraborty, Kagwani, & Dey, Performance Comparison of Incremental K-means and Incremental DBSCAN Algorithms, 2011). Lastly, the term “dynamic” was used by (Jain A. K., Data clustering: 50 years beyond K-means, 2010), to describe clustering of data as it changes over time, with “streaming” being a specific type of dynamic data. In contrast, (Han & Zhao, 2005) use the term “dynamic” to mean that their clustering algorithm can determine the number of clusters from the static data set without being user specified. While the terms online, incremental, streaming, evolutionary, adaptive, dynamic, and single or one pass have all be used in various ways to describe clustering that can occur as data enters a system, the adjective “online” will be adopted for the remainder of this document.

A discussion of four common partitioning clustering algorithms follows. Three of the four have seen successful use in clustering in HSI. The final algorithm (DBSCAN) does not appear to have been used to cluster pixels in HSI applications, but it is listed because specific extensions to the algorithm embody concepts that are useful for this project.

K-means

K-means is one of the most widely used and referenced heuristic algorithms for clustering (Jain A. K., Data clustering: 50 years beyond K-means, 2010). The algorithm requires that the number K (the number of clusters) be specified before the algorithm executes. At initialization, K exemplars are selected to be candidate cluster centers (all K clusters are formed). All exemplars are then placed into the nearest clusters using some distance measure (such as Euclidean). In some implementations, the centroids are moved as each point is added, as in the “online” option in k-means in (Mathworks, 2012). In other implementations, the centroids are only recalculated after all exemplars have been clustered as in the “block” processing option in (Mathworks, 2012). The algorithm repeats until the cluster centroids do not move following a complete iteration through the exemplars. As this is a heuristic, seeking to minimize the total sum of the squared distances between each exemplar and its associated cluster center, the algorithm is not guaranteed to find a global optimum, but if it is run to convergence, it will find at least a local optimum (Duda, Hart, & Stork, 2001). By minimizing a single sum of squares measure, with equal weight to all dimensions in all directions, a Euclidean distance will result in the algorithm seeking circular, spherical, or hyperspherical clusters. In an effort to find the global (or better) optimum, the algorithm is often repeated, with the “best” run

being taken. Best could be defined as lowest total variance. A total of five repetitions was found to be adequate for HSI applications using correlation distances by (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007). K-means is described by (Duda, Hart, & Stork, 2001) as having an order complexity of $O(pdkT)$ where p is points, d is dimension, k is the number of clusters and T is the number of iterations.

ISODATA

The Iterative Self-Organizing Data Analysis Techniques A, or ISODATA, was created to address pattern recognition problems (Ball & Hall, 1965). Where the k-means algorithm requires the number of clusters to be set by the user, this technique performs both splitting and merging of clusters during processing according to specified parameters, allowing the number of clusters to be determined by the algorithm. The standard deviation in a cluster (using some distance measure) exceeds a specific threshold, the cluster is split. If two clusters, become closer than a specified distance (using the distances between cluster centroids), they are combined. The specification of these two thresholds is critical to the performance of the algorithm. When evaluated for multi-spectral image processing, its performance was found to be inadequate when using transformed divergence distances (Cobb, 1988). When evaluated for the purposes of clustering in HSI, its performance was found to be reasonable, but sensitive to the specific algorithm parameter settings using Euclidean distance (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007). Based on its successful previous use for clustering in multispectral images from IKONOS and hyperspectral images from Hyperion, ISODATA was successfully used in conjunction with other

techniques including the use of spatial information for image segmentation in the case where the number of clusters was reasonably well known *a priori* (Tarabalka, Benediktsson, & Chanussot, 2009), and limits on the total and minimum number of clusters were set. As with K-means, a Euclidean distance with ISODATA would be expected to produce circular, spherical, or hyperspherical clusters.

X-means

A different approach to automatically determining the appropriate number of clusters was taken by (Pelleg & Moore, 2000) in the X-means algorithm. Where ISODATA uses specific thresholds to combine or split clusters, X-means relies on the Bayesian Information Criteria (BIC), as formulated by (Kass & Wasserman, 1995) to determine if clusters should be combined or split. The calculation used in the original paper is shown in Equation 2.

$$BIC = \frac{-n_i}{2} \ln(2\pi) - \frac{n_i d}{2} \ln(\sigma^2) - \frac{n_i - K}{2} + n_i \ln(n_i) - n_i \ln(n) \quad \text{Equation 2}$$

(2)

Where:

n_i = number of points in cluster i

n = total number of points in clusters under consideration

d = dimensions in use

K = number of total clusters under consideration

It is important to note that a single sigma value is used in this calculation and an identical spherical Gaussian assumption was used in the derivation. Further, the sign convention used in this BIC calculation differs from some other sources (Kutner, Nachstheim, Neter, & Li, 2005), such that a greater BIC is preferable when comparing to models for greater parsimony and explanatory power.

The algorithm performs two main functions, named “Improve-Parameters” and “Improve Structure” in the original paper. Improve-Parameters is simply running K-means to convergence. In Improve-Structure, some existing clusters are split using K-means with K set to two. The BIC for the points in a single cluster is compared to the BIC of the points split into two clusters. The winning BIC (the larger BIC in this formulation) determines the new structure. An implementation using k-means with correlation distances by (Williams, Robustness of Multiple Clustering Algorithms on Hyperspectral Images, 2007) showed adequate performance for a relatively low upper limit on cluster numbers. As with K-means, the algorithm is expected to seek circular, spherical, or hyperspherical clusters using Euclidean distances.

DBSCAN

A density based method, for clustering data into irregular cluster shapes was developed by (Ester, Kriegel, Sander, & Xu, 96). Two parameters define the behavior of the algorithm. The first, Eps, is a single radius that defines the neighborhood around a point. In the original formulation, a single value of Eps is used. The second parameter, MinPts, defines the minimum number of points inside Eps that constitute a sufficient density for the points to be inside of the same cluster. Because the notion of being in the same cluster is not tied to a distance from a single point (unlike K-means or X-means), the clusters may be of arbitrary shape.

The survey by (Xu & Wunsch, 2005) indicates that DBSCAN is not suitable for high dimensionality data. This may explain the absence of DBSCAN from HSI processing literature. Despite its apparent lack of use in HSI, several extensions to DBSCAN have particular relevance to online clustering in HSI. The incremental

DBSCAN algorithm developed by (Chakraborty & Nagwani, Analysis and Study of Incremental DBSCAN Clustering Algorithm, 2011), performs the DBSCAN algorithm as data enters the system. The key logic in the algorithm (similar to other online algorithms and Looneys' high level algorithm) is the placement of the new sample into the either an existing cluster (based on the density distance criteria in DBSCAN), or into an outlier status if it cannot be placed into an existing cluster based on the density criteria (e.g. a threshold). If enough outliers occur in the same region, those become a new cluster. Every new point may be placed in a cluster, be declared an outlier, or be the final element that turns a set of outliers into a cluster as the data are collected.

A local density version of DBSCAN, named LDBSCAN, was developed by (Duan, Xu, Liu, & Lee, 2009) to vary the single density estimate encoded in the Eps and MinPts parameters, by cluster. This paper also contains a major contribution in the form of an explication on the various available definitions of an outlier, and a method whereby a small cluster (from a total membership perspective), with a low density relative to other clusters may be considered itself an outlier. In short, LDBSCAN demonstrates that the variation in each cluster may differ, and outlier detection can occur as a byproduct of clustering, even when there are sufficient outliers to be placed into a cluster together.

Summary

Recent research highlights the utility of subspace clustering using the principal components of the spectral intensities in HSI for anomaly detection. Operational considerations warrant an efficient, on-line approach to anomaly detection, which necessitates an efficient online clustering algorithm. While numerous online clustering

algorithms exist, none are currently tailored to the particular requirements of the HSI domain.

Given that the Mahalanobis distance in a principal component subspace can be used in a detector to determine whether or not a pixel is the member of the background (a collection of similar pixels), and that clustering can subdivide the background for more sensitive use of such a detector, it is possible that using the Mahalanobis distance to determine cluster membership as an image is collected might form the basis of a valid clustering algorithm, which may, as a byproduct, separate anomalies from the background clusters, essentially behaving similarly to an ILRX detector supported by clustering. In fact, if the members of various clusters can be separated from each other by means of the Mahalanobis distance, a single pass clustering method based solely on Mahalanobis distance should be viable.

III. Methodology

Chapter Overview

This chapter first describes the imagery data set used to develop and assess the algorithm. Next, a set of preliminary experiments to be used to confirm the expected desirable performance of Mahalanobis over Euclidean distances in the principal component subspace for the online clustering algorithm are described. The structure of the developed algorithm and key equations required for efficient implementation are then presented, assuming the use of Mahalanobis distances. Finally, a description of the methods to be used in evaluating the performance of the algorithm are given.

Image Data Set

The imagery used for initial experimentation, algorithm parameter selection, and validation was collected under the program described by (Rickard, Basedow, Zalewski, Silvergate, & Silvergate, 1993). The imagery includes the visible spectrum through approximately 2.5 μm , which (Hobbs, 2000) describes as “near infrared, defined roughly as the region in which glass lenses work and decent photodiodes are available.” The available images have both a forest and dessert background, and contain various targets. The images do not have associated ground truth on a pixel by pixel basis, which would otherwise allow for a numerical assessment as to whether or not the clusters formed by the algorithm correspond directly to different signatures or classes of signatures, but an anomaly mask, which indicates for each pixel whether it is or is not an anomaly, is available.

Preliminary Experiments on the Distances Measures in the Principal Component Subspace

The following tests each provide some characterization of the behavior of Euclidean or Mahalanobis distances in the principal component subspace. The condition number and clustering tests provide information on clustering behavior using Euclidean distances. The Eigen value and loadings tests will give an indication as to whether or not the content of the image might drive the calculation of the principal component scores in a way that could complicate a clustering algorithm using such scores. Finally, the distance tests will indicate the relative stability or instability of the Mahalanobis and Euclidean distances in the principal component space.

Condition Number Test

The condition number of a matrix can be defined several different ways. The ratio of the largest to the smallest Eigen value of the matrix will be used here. Using this definition, the condition number of a covariance matrix can be interpreted as the ratio of the largest variance seen in any direction in the space to the smallest. For the Euclidean distance to be a valid distance for clustering, clusters should be roughly spherical, resulting in the condition number having a value of one when calculated using the pixels in any given cluster.

It is possible that the number of clusters or the number of dimensions used to form clusters could impact the degree to which the formed clusters are spherical. By running the k-means algorithm on a typical image from the available set with a wide range of values for both number of clusters and retained principal components, any dependence on either of these factors should be visible in the condition numbers of the

clusters. If the number of clusters is large, it might be difficult to assess the overall impact of the factors on the condition numbers, but a three point approximation of the distribution of the condition numbers in each case could provide an easily understandable reading as to the impact. The maximum, minimum, and median values are natural candidates for the three points. These values could be readily displayed in a color coded two dimensional image. Values substantially different from one in various combinations of retained principal components and numbers of clusters would indicate that Euclidean distance would be inappropriate.

Clustering Test

Viewing the results from a k-means clustering using Euclidean distance for a low number of retained principal components and a small number of clusters could provide a visual indication of the adequacy or inadequacy of the Euclidean distance. The scores for two principal components can easily be displayed in a plot. If some small number of natural clusters appear to the eye, this number could be used to run k-means with the Euclidean distance to determine if the algorithm can find the natural clusters. Color coding in image natural colors could be used on the scores to determine if the natural distribution of objects in the image might be easily clusterable. Color coding could also be used to show the clustering of the points, with a unique color for each cluster. Undesirable or inadequate behaviors in the clustering would include splitting contiguous regions or including discontinuous regions in the same cluster.

Eigen Values by Signature Test

The Eigen values of the covariance matrix used to from the principal components reflect the amount of variance present in each associated principal component axis. A

plot of the Eigen values is often the first step in determining the dimensionality of the data set, and resultant number of principal components to retain for further processing. By producing this plot for collections of pixels that contain the same, common, background elements such as trees, roads, sand, or fields, it should be possible to determine whether or not the backgrounds in the image might change the dimensionality, and potentially drive a different number of retained principal components for different images. Similarly, plotting the Eigen values of the covariance matrix for a collection of random known anomalies would provide an indication as to whether the presents of anomalies might also change the number of retained principal components.

Eigen Values by Combined Signatures Test

As in the Eigen value by signature test, placing pixels from various homogeneous backgrounds together into a single set for principal component analysis and Eigen value by principal component plot would give an indication as to whether combinations of common background items together might influence the dimensionality above any individual influence. The addition of the anomalies in differing amounts, such as 1% and 10%, would give an indication as to whether the amount of anomalies in the image might influence the dimensionality.

Loadings of Paired Signatures

The loadings matrix in principal components analysis provides, for each principal component, the correlation between the original variables and the principal components. This information can give insight into which original variables drive each principal component response. If the presence of distinct backgrounds in an image influence the principal components by greatly increasing the overall variation in the data set, it would

be expected that one of the first few principal components might have a loading pattern that reflects the mean difference between the backgrounds in the original variables. By placing two distinct types of background pixels into the same set for principal component analysis, and calculating both the loadings and the mean difference by band in the original data set, the values across the bands can be compared to determine if the mean difference pattern is reflected in the principal components.

Distance Plots

By placing known homogenous pixels into predefined clusters, it is possible to calculate a Mahalanobis and Euclidean distance for all the pixels in the set to each of the cluster centroids. This can be done across a range of retained principal components as an additional check on the dimensionality assessment. If either distance shows a clear distinction between pixels inside and outside the cluster at some number of retained principal components, then a threshold could be set for use in a clustering algorithm. The addition of randomly selection anomalies to the set will allow an assessment as to whether the presence of anomalies in an image might impact a set distance threshold. The algorithm will be developed with the assumption that a threshold of this nature exists.

Algorithm Description

This section covers the high level structure of the algorithm to provide the needed context to describe the *initialization*, *cluster assignment*, cluster merging and cluster splitting portions of *improve structure*, and *anomaly detection* steps. Although cluster splitting precedes cluster merging steps in the structure improvement portion of the

algorithm, cluster merging is presented first, as it more efficiently provides the information that forms the basis of the cluster splitting step.

High Level Structure

The overall structure of this online algorithm follows the structure of Looney's high level algorithm, with the addition of a modified improved structure step taken from the X-means algorithm. Similarity to existing clusters is assessed in the step labeled classify new data, and is performed by using a Mahalanbois distance to the existing cluster centers as the similarity measure. A flow chart describing the steps of the algorithm appears in Figure 4. A rudimentary anomaly detector is included in the algorithm to demonstrate both the ability to detect anomalies while processing the image and the concept of declaring anomalies based on the numbers of members in clusters. The detector does not affect the clustering behavior.

Two light gray areas in the chart correspond to steps that are either not fully implemented, or only implemented for demonstration rather than complete evaluation. The step to remove old data from clusters is only implemented for counts to support the detector. The split cluster step under improve structure is implemented, but only to demonstrate the functionality. The computational complexity of the step as implemented makes this function prohibitive to perform at every step as shown in the diagram.

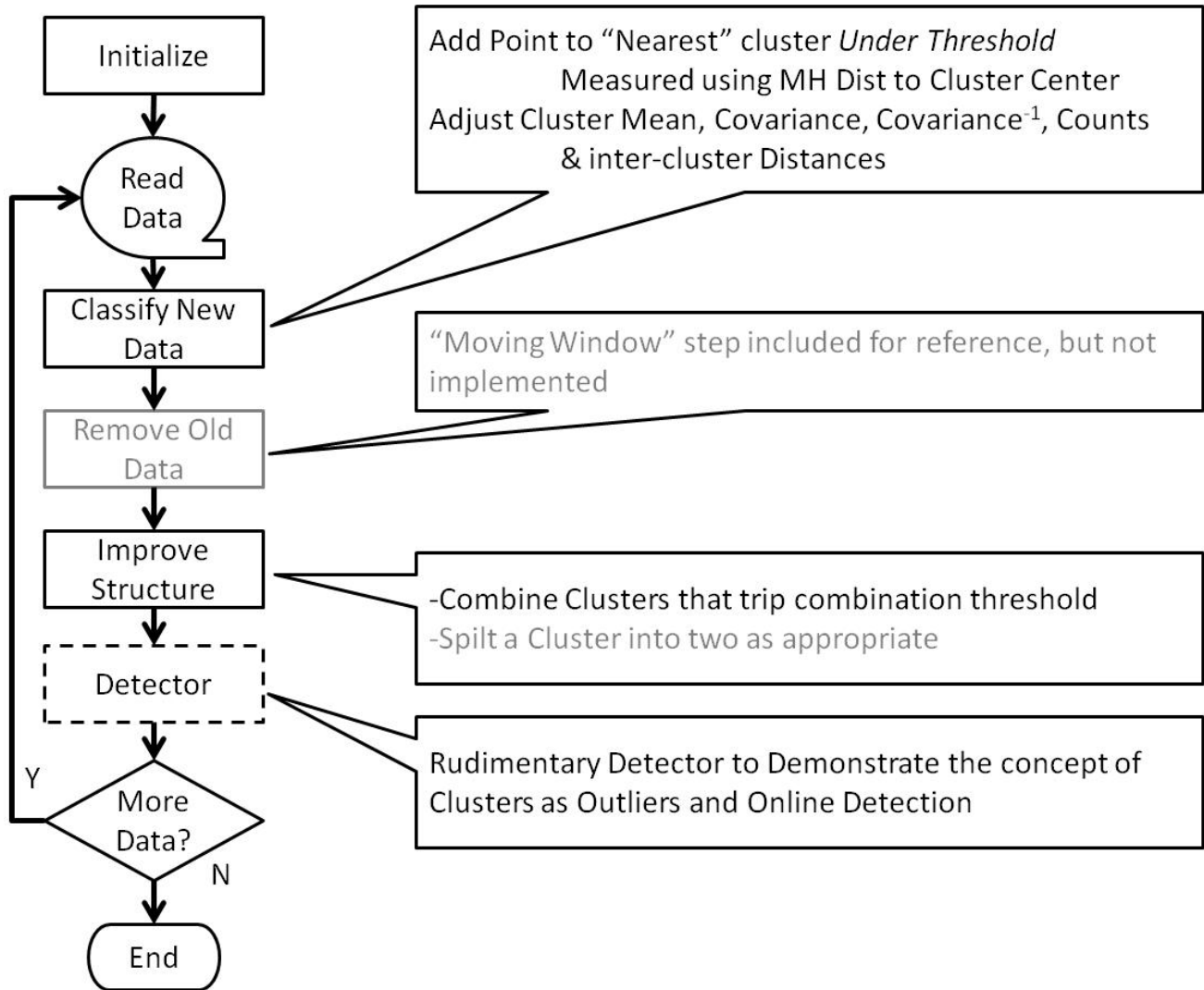


Figure 4: Online Clustering Algorithm Flow Chart

Two global data structures are required for the individual algorithm steps described below. First is a table of distances between each cluster. The other is a running record indicating to which cluster each pixel has been assigned. The anomaly detection portion of the algorithm also maintains a list of which pixels have been declared anomalous.

Initialization

The initialization step includes three actions required to start the algorithm. Specifically, band selection and dimensionality reduction to create the subspace for clustering using principal component methods, and the formation of the first cluster are included in initialization.

Band Selection

In the manner of (Johnson, 2008), it is assumed that the defective, problematic, and absorption bands of a sensor are known and discarded before processing. The images used in this project contained 210 available bands, but bands numbered 1-9, 98-114, 133-157, and 201-210 were excluded from processing. Figure 5 illustrates the noise issue present in the first several bands and the behavior of the sensor in the atmospheric absorption bands. The spectrum plot is useful for visually finding bands with little to information, but a review of an images formed from information from a single band, for each band, is required to visually determine if there were any issues that impact the system spatially and only in a single band.

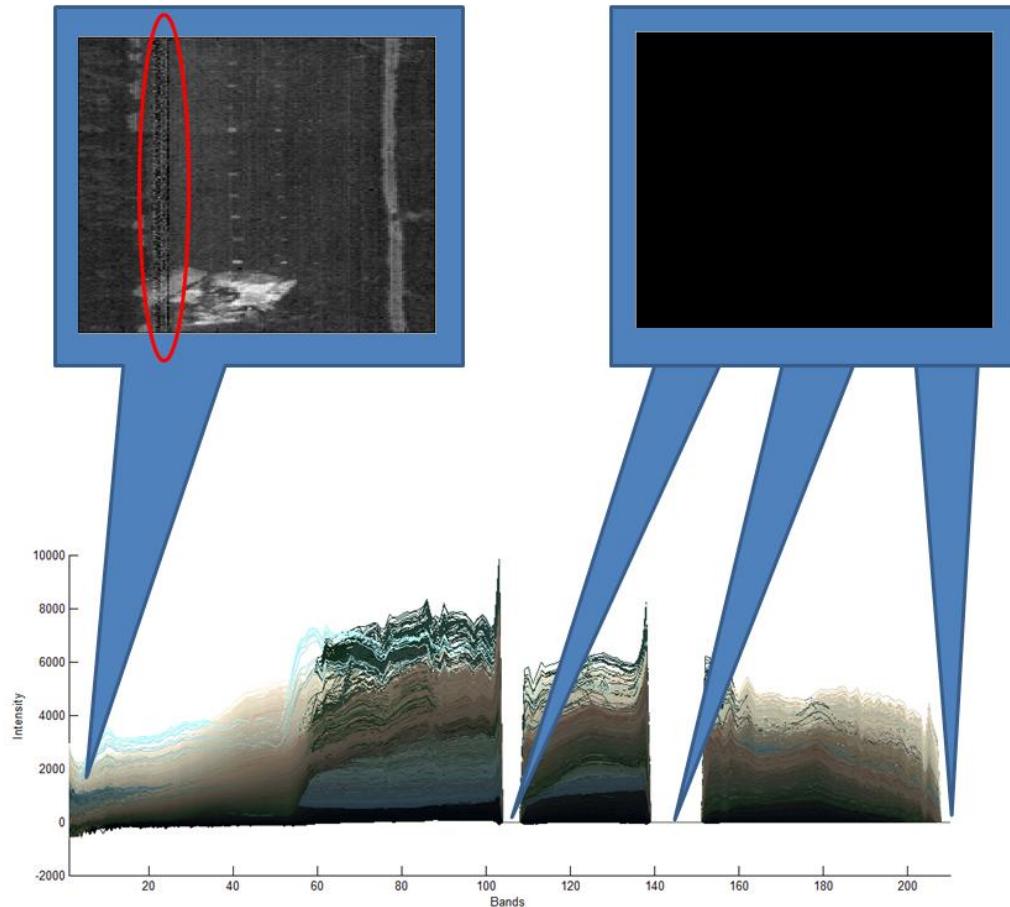


Figure 5: Spectra of Hyperspectral Image with Discarded Bands Exploded

Principal Component Dimensionality Reduction

Dimensionality reduction to form the sub-space for the clustering algorithm is accomplished for this exercise by performing the principal component analysis on the entire image after the removal of the noise and absorption bands prior to processing. While this would not be possible in an online application, implementation of online-principal components analysis in addition to online clustering is beyond the scope of this effort. It is assumed that an online principal component algorithm or other mitigations, such as a single set of principal components for all images, would be sufficient to address

the concerns raised by (Bush, 2012), relating to the possible changes in principal component behavior across an image.

Initial Cluster Formation

For the formation of the first cluster, the algorithm reads in two pixels, and forms a covariance matrix. A test is then performed to determine whether the covariance matrix is valid matrix. Specifically, the number of points used to form the covariance matrix must be at least the number of dimensions in use plus one. Further the covariance matrix must be well conditioned, to allow for stable calculations and inversion. With the computer and Matlab installation used for experimentation (Mathworks, 2012), a minimum value of 10^{-16} for the reciprocal of the condition number of the matrix was found to be suitable. An additional test for sufficiency of the covariance matrix was to ensure that the value of the determinant, also known as the generalized variance, was positive. The initial loop continued to read in pixels from the image until all three of these criteria were met to form the initial cluster.

The initial cluster is indistinguishable from all other clusters that may be found by the algorithm. Specifically, all the items described in Table 1, are set for the initial cluster, updated in subsequent steps of the algorithms as required, and defined for all subsequent clusters as they are formed and updated.

Table 1: Elements of a Cluster

Name	Description
M	Centroid vector for the cluster (mean of the member pixels
S	Covariance matrix
S^{-1}	Inverse of S, calculated and stored to improve computational efficiency.
N_{current}	Number of pixels currently assigned to the cluster
Enuf	Flag set to indicate that sufficient points are in the cluster to allow the creation of a well conditioned covariance matrix

Mbr	List of pixels which are members of the cluster
N _{detection}	Number of pixels in the cluster within the recent memory defined for the detection portion of the algorithm

With regard to the selection of pixels in the cluster, two possible errors could have occurred at this point in the algorithm. First, elements from more than one cluster may have been placed into the initial cluster. This is addressed through the split cluster step of improve structure discussed below, which will have the opportunity to split this cluster into the appropriate parts in future processing. Second, the first few elements may be closely correlated (due to natural spatial correlation) leading to a covariance matrix with a lower variance in any number of dimensions than is actually representative of the cluster. This would result in several clusters containing elements from what should be the same cluster (e.g. extra clusters). This is addressed by the merge cluster step of improve structure described below. The approach of forming a single initial cluster during initialization was selected over the competing option, namely running an existing clustering algorithm such as k-means, x-means, or ISODATA to form a set of initial clusters, due to its simplicity. Running k-means would require the selection of both the number of initial clusters and the number of points to cluster. Notionally, the X-means algorithms would automatically select the number of clusters, but the number of initial points to consider would still need to be defined. Also, the assumption that the first few pixels collected on the edge of the image are likely to be from a relatively homogeneous region or the same object seems to be well founded based on a cursory inspection of the available HSI data sets.

Without regard for the generating the principal components, or the number of principal components retained, the computational complexity of the initialization step is $O(p_1)$ where p_1 is the number of pixels required to form a well conditioned covariance matrix for the first cluster. At each step, a covariance matrix is calculated. This has a complexity of $O(d^2 p_i)$ where p_i is the number of pixels used in the calculation at each step and d is the dimensionality or number of retained principal components. This yields a total computational complexity for this step of $O(d^2 p_1^2)$. If it is expected that p_1 will be only slightly more than d on average, and if the algorithm were modified to read in d pixels initially to skip the first d iterations of the loop, this step could be reasonably expected to occur with complexity $O(d^3)$. This also happens to be the computational complexity of calculating the determinant, which is also required for this step. This step is executed only once during the algorithm, and is consequently not a major driver to the computational complexity of the entire algorithm. The calculation of the Eigen values to check the covariance matrix is not considered.

Cluster Membership Selection

Following initialization, each subsequent pixel is read and assigned to a cluster. As this is a distance thresholding algorithm, cluster membership for each pixel is based on the distance to the existing clusters. A pixel is assigned to the nearest cluster, unless that distance is greater than a specified threshold, in which case the pixel is assigned to a new cluster containing only itself.

To allow for hyperellipsoidal clusters, the Mahalanobis distance is primarily used. The calculation of the Mahalanobis distance is shown in Equation 1. For computational efficiency, the square of the Mahalanobis distance can be used as a threshold, as the

square root is a monotonic function, and only the comparison between the distance and the threshold is important.

Figure 6: Cluster Membership Determination, illustrates the motivation behind both a distance threshold and the use of Mahalanobis Distance over Euclidean distance. Candidate point 1, shown as a purple diamond, is closer to the first cluster in terms of Euclidean distance, but visually, it appears that it should be a member of cluster 2. The Mahalanobis distance to cluster 1 is approximately 5, while the Mahalanobis distance to cluster 2 is less than 2. Meanwhile, candidate point 2, shown as the red diamond, is not close enough to either cluster to warrant assigning it to an existing cluster.

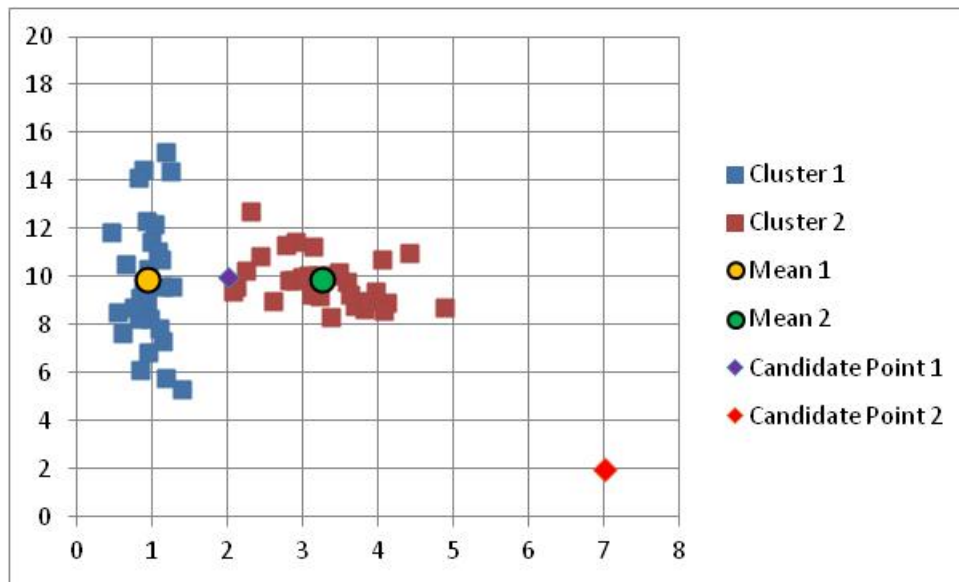


Figure 6: Cluster Membership Determination (Mahalanobis Distance Motivation)

Any domain exhibiting the behavior shown in Figure 6 would require the use of Mahalanobis over Euclidean distances for appropriate cluster. While this does not discount the use of other distances measures (such as divergences), for these two options, the Mahalanobis distance clearly is preferable if the pixels are expected to have a

hyperellipsoidal distribution. The results of the initial experiments in Chapter 4 should show whether or not Mahalanobis distances are, in fact, preferable.

The Mahalanobis distance calculation requires the inverse of the covariance matrix for each cluster. The construction of the first cluster during initialization guarantees the availability of a covariance matrix and its inverse, but once a single pixel falls outside the set threshold for cluster membership for the existing clusters, a singleton cluster is formed. As discussed in the initialization section, a single point cannot have a covariance matrix. In fact, in n dimensions, at least $d+1$ points are required to build a covariance matrix. Further, ill conditioning can still be an issue with only $d+1$ points used in an d -dimensional space. When a singleton cluster is formed, the *Enuf* flag for the cluster is set to 0, indicating that a covariance matrix and its inverse are not available for the cluster.

The existence of clusters with no covariance matrix presents a problem, in that by using the Mahalanobis distance alone, no second point would ever be added to the singleton clusters because no Mahalanobis distance can be calculated between a pixel and those clusters. To address this issue, a set of four Euclidean distance based rules are considered. This requires the calculation of Euclidean distances simultaneously with the Mahalanobis distances.

In all cases, if the nearest cluster from a Mahalanobis distance perspective is also the nearest cluster from a Euclidean perspective, the pixel is assigned to that cluster, unless the Mahalanobis distance threshold is exceeded, in which case a new cluster is formed. In all cases, if the nearest cluster from a Euclidean perspective contains sufficient points to calculate a Mahalanobis distance, but another cluster is closer from a

Mahalanbois distance perspective, the nearest Mahalanbois distance cluster is selected, unless the Mahalanbois distance threshold is exceeded, in which case a new cluster is formed. If the nearest cluster, from a Euclidean distance perspective, does not have enough points for a valid covariance matrix, the four rules direct that the pixel be added to the nearest cluster, from a Euclidean perspective, unless a set threshold has been exceeded. The threshold differs for these four rules.

For the first rule, there is no threshold. Every pixel will be assigned to the nearest cluster from a Euclidean perspective if it does not contain enough points to have an associated covariance matrix. Rule two behaves the same as rule one, except in cases where the distance to the cluster is greater than the minimum Euclidean distance between the centroid of the nearest cluster and the centroid of the cluster's nearest neighbor as measured by Euclidean distance. Rule three is the same as rule two, except that the threshold is determined by the median of the Euclidean distances between the centroid of the nearest cluster using Euclidean distances and all other cluster centroids. The fourth rule uses a threshold of the maximum Euclidean distance between the centroid of the nearest cluster and all other cluster centroids.

Figure 7 illustrates the application of these rules in two dimensions. A candidate pixel may only be assigned to the nearest cluster, so the existing clusters create a Voroni partitioning of the space as shown by (Pelleg & Moore, 2000). This is illustrated by the black equidistance line in this simple example. The thresholds used by rules 2 through 4 are illustrated by the dashed circle around cluster 1. Candidate point 1, shown as a purple diamond, is closest to the first cluster, on the left of the equidistance line, and within the threshold determined by the spacing of these two clusters. This candidate would be

assigned to cluster 1. The second candidate point, illustrated by a red diamond, is closest to the first cluster, but outside the threshold, so it would be assigned to a cluster alone under rules 2 through 4. For rule two, cluster 2 would be the nearest cluster to cluster 1. For rule 3, cluster 2 would be the cluster with the median distance from cluster 1. For rule 4, cluster 2 would be cluster farthest away from cluster 1. For rule 1, even candidate point 2 would be assigned to cluster 1. Each of the four rules will be explored to determine which is the most appropriate.

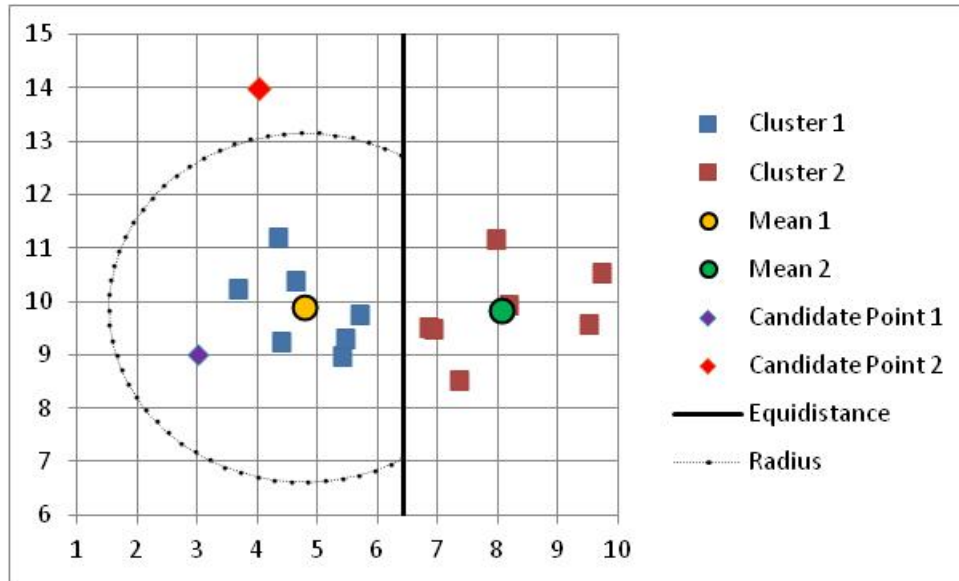


Figure 7: Cluster Membership Determination (Euclidean Rules Two through Four)

Once a pixel has been assigned to a cluster, all elements from Table 1, must be updated. Fortunately, updates to the elements of the table can be performed without direct reference to the individual members of the clusters. The new centroid is simply a weighted average of the existing centroid and the new pixel where the weights are determined by the number of pixels in the cluster. Similarly, the covariance matrix can be updated by way of Equation 3. The fact that the updates do not require a calculation

involving all previous pixels added to the cluster saves considerable computation time as the number of pixels in each cluster becomes large.

$$S_{i'} = \frac{(n_i - 1)}{n_i} S_i + \frac{1}{n_j} (x_i - \bar{x}_{old})^t (x_i - \bar{x}_{new}) \quad \text{Equation 3} \quad (3)$$

Where:

n_i = number of points in cluster i prior to the update

S_i = sample covariance matrix for cluster i prior to the update

\bar{x}_i = n dimensional mean vector for cluster i

\bar{x}_{old} = n dimensional mean vector for cluster j prior to the update

\bar{x}_{new} = n dimensional mean vector for cluster j subsequent to the update

t indicates transposition

For clusters without a covariance matrix, the addition of a point initiates a check, as was done for the first cluster during initialization, to determine if a valid covariance matrix is possible with the additional point. If a valid covariance matrix is found, the Enuf flag is set to 1.

This step of the algorithm requires little more than a calculation of two distances to each existing cluster, and a potential search of the list of distances between all the clusters and a single cluster in the case of three of the four Euclidean rules. For pixels added to clusters under the Euclidean rule, additional work is required to calculate a covariance matrix, as in the initialization step, which has already been assumed to be $O(d^3)$. The search, without regard to the matrix calculations at each step, has order complexity is $O(c)$ where c is the number of clusters. The computational complexity of matrix multiplication, and consequently matrix inversion, is currently possible in $O(d^{2.376})$ and never better than $O(d^2)$ per (Cormen, Leiserson, & Rivest, 1998), but $O(d^3)$ is often used as the practical limit. This ultimately gives this step a computational complexity of $O(cd^3)$.

Remove Old Data

While not fully implemented, the step to remove old data would function like the cluster update, but in reverse. The pixel at the end of the recent memory would be removed from the count of elements in the cluster, the membership list, and its contribution would be removed from the covariance matrix and the centroid. The length of the memory would be a user defined parameter. In a real time system this might be a function of available memory in the system. The current implementation does not remove the contribution of any pixel from the covariance matrices or the centroids, but a separate set of counts is maintained for use by the detector alone. The current implementation of this step in the algorithm serves only to support the detector, and has no impact on the clustering. Once a pixel is read into the algorithm, it remains active in the algorithm for all centroid and covariance calculations for assignment, and improve structure steps.

Improve Structure (Cluster Merging)

As mentioned previously, it may be necessary to combine existing clusters into a single cluster. This case is illustrated in Figure 8. The first few points happened to be unusually close together, relative to the overall structure of the cluster, and create a very small covariance. If various separate parts of the image have a similar signature, and belong in the same cluster, but have high spatial correlation when they are read in order, this behavior might not be unexpected. A nominal point in the cluster read in after the formation of the inappropriately “tight” covariance cluster might be have a distance greater than threshold and separated into a new cluster. As more points, shown in blue, are added to the two clusters, it becomes visually apparent that the two clusters are

actually as single cluster. It should be the case that chances of this error occurring should be lowered with higher dimensions, as more points would be required to form the covariance matrix, depending on the behavior of the Euclidean rule in adding points to clusters, but for computational efficiency, and to be discussed later, appropriate detector performance, the clusters should be combined.

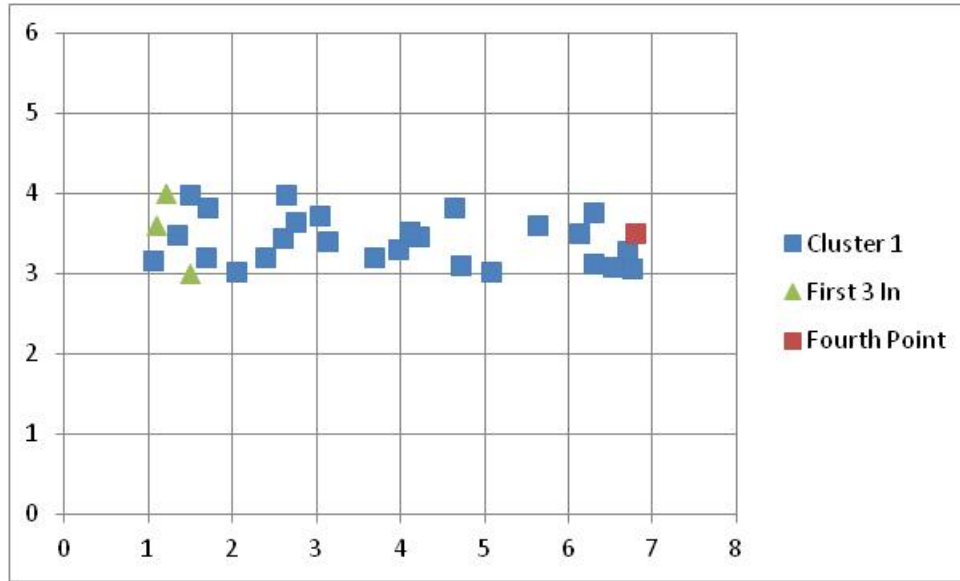


Figure 8: Potential Error Addressed by Combining Existing Clusters

ISODATA combines clusters when their Euclidean distance falls below a specific threshold. While the Euclidean distance between the clusters offers a convenient number, using Euclidean distances would require setting a user defined threshold. Further Euclidean distances do not take the hyperellipsoidal nature of HSI clusters into account when forming the distance. The Bhattacharyya distance is an analog to the Mahalanobis distance for use with two clusters of data. The calculation for Bhattacharyya distance is shown in Equation 4.

$$D_{BH} = \frac{1}{8} (x_i - x_j) \left(\frac{S_j + S_i}{2} \right)^{-1} (x_i - x_j)^t + \frac{1}{2} \ln \left(\frac{\left| \frac{S_j + S_i}{2} \right|}{\sqrt{|S_j| |S_i|}} \right) \quad \text{Equation 4} \quad (4)$$

Where:

S_i = sample covariance matrix for cluster i

S_j = sample covariance matrix for cluster j

x_i = centroid for cluster i

x_j = centroid for cluster j

The calculation requires the inverse and the determinant of a new matrix which is formed from the arithmetic average of the elements of the two cluster's covariance matrices. This new matrix would be used only for the purposes of this calculation. While this can be computed without reference to the individual elements in the clusters, it is still computationally expensive, particularly when this calculation would need to be performed once for each cluster in the distance table when a single cluster's position is updated. If used, a distance threshold would still be required to determine when two clusters should be combined. These facts likely account for the absence of this distance measure from the clustering literature.

X-means offers a more computationally efficient method of assessing whether or not two clusters should be combined through the BIC calculation comparing the BIC of the combined clusters to the two single clusters. One potential weakness in the BIC used in X-means is the assumption of a single variance. A BIC formulation with potentially greater applicability to this specific clustering problem with differing variances and covariance's for each cluster is given by (Biatov, 2010) in the form of the delta BIC shown in Equation 5.

$$\Delta BIC = \frac{n_i \ln |S_i|}{2} + \frac{n_j \ln |S_j|}{2} - \frac{n_{ij} \ln |S_{ij}|}{2} + \lambda P \quad \text{Equation 5} \quad (5)$$

Where:

n_i = number of points in cluster i

n_j = number of points in cluster j

S_i = covariance matrix for cluster i

S_j = covariance matrix for cluster j

n_{ij} = number of points in the combined cluster ij

S_{ij} = covariance matrix for the combined cluster ij

λ = adjustment parameter

P = penalty term

The penalty term, which is the only factor expressly accounting for the dimension of the space, is shown in Equation 6.

$$P = 0.5(d + 0.5d(d + 1)) \ln (n_i + n_j) \quad \text{Equation 6} \quad (6)$$

Where:

n_i = number of points in cluster i

n_j = number of points in cluster j

d = number of dimensions (n_{pc})

The delta BIC is simply the difference in the BIC's for the one cluster and two cluster case. As formulated, a positive value indicates that a single cluster is preferred over two. The use of a single number with a clearly defined threshold allows the delta BIC to be used as a distance by the algorithm. The calculation here differs from the one from X-means mainly in the use of the generalized variance, rather than a single standard deviation estimate for the variation in the clusters. This should provide a better assessment for the HSI domain. Because the covariance matrices for the individual cluster members are stored as the algorithm proceeds, the BIC can be calculated without reference to the individual elements in the clusters. Although a new matrix is required by

the calculation, an inverse of a new matrix is not required, as would be required if Bhattacharyya distance were used. The new matrix that is required can be calculated using the relationship defined in 7.

$$S_{ij} = \frac{(n_i - 1)S_i + n_i \bar{x}_i^t \bar{x}_i + (n_j - 1)S_j + n_j \bar{x}_j^t \bar{x}_j - (n_i + n_j) \bar{x}_{ij}^t \bar{x}_{ij}}{(n_i + n_j - 1)} \quad \text{Equation 7} \quad (7)$$

Where:

- n_i = number of points in cluster i
- n_j = number of points in cluster j
- S_i = sample covariance matrix for cluster i
- S_j = sample covariance matrix for cluster j
- S_{ij} = sample covariance matrix for a cluster composed of clusters i & j members
- n_{ij} = number of points in the combined cluster ij
- \bar{x}_i -bar = n dimensional mean vector for cluster i
- \bar{x}_j -bar = n dimensional mean vector for cluster j
- \bar{x}_{ij} -bar = n dimensional mean vector for the combined cluster ij
- t indicates transposition

Efficiency can be further improved by using the relationship shown in Equation 8.

$$(n_i + n_j) \bar{x}_{ij}^t \bar{x}_{ij} = \frac{(n_i \bar{x}_i + n_j \bar{x}_j)^t}{(n_i + n_j)} \quad \text{Equation 8} \quad (8)$$

Where:

- n_i = number of points in cluster i
- n_j = number of points in cluster j
- \bar{x}_i = n dimensional mean vector for cluster i
- \bar{x}_j = n dimensional mean vector for cluster j
- \bar{x}_{ij} = n dimensional mean vector for the combined cluster ij
- t indicates transposition

As implemented, the entire inter-cluster distance table need not be searched on each combination step. As a cluster is added, a column is added to the delta BIC distance table. Until the cluster has sufficient points to form a covariance matrix, no action is taken in this column. When the final pixel required to form the covariance matrix is

added, the column is populated with distances. If any of these distances are below the combination threshold, the cluster will be combined with another cluster. If multiple distances are below the threshold, the cluster will be combined with the closest cluster. Once the combined cluster is formed, the single column (and/or row) is updated in the same manner, checking for distances below threshold. This continues until the modified column does not contain any distances which violate the threshold. When the distance table is first formed with the initial cluster, there are no distances. The addition of all succeeding clusters is done to prevent a distance which violates the threshold from remaining in the table. By induction, there can be no distances which violate the threshold in the table.

The cluster merging step is conducted after cluster splitting. While cluster splitting can only occur once per new pixel, a merged cluster could be close enough to another cluster to require an additional merge. Further, if a cluster is split into two clusters in the cluster splitting step, either one of both of the new, smaller clusters could be close enough to other clusters to require merging. Consequently, the merging step is allowed to repeat for as long as the distance table contains positive delta BIC values. For the case where a cluster has been previously split, both clusters are checked, with the cluster with the greatest violation of the distance threshold being combined first.

Combining two clusters requires the new cluster's distances to the other cluster's to be updated. With the required matrix multiplications having a complexity of $O(d^3)$, for c clusters, the resulting computational complexity is $O(cd^3)$ for a single combination. While it is possible that $c-1$ combinations could occur, resulting in a complexity of $O(c^2d^3)$, such an event is considered unlikely, and would result in a single cluster for the

next step. The expected number of iterations through this step for a single pixel is one resulting in an expected complexity of $O(cd^3)$.

As implemented, an additional step occurs when two clusters are combined. Cluster membership lists are implemented in arrays rather than linked lists, implying the complexity of the combination step becomes $O(p_i)$ where p_i is the number of pixels to be copied into combined cluster's membership list. This is also true if the data structure which lists the associated cluster for each pixel is updated.

The algorithm was implemented with the ability to disable cluster merging.

Improve Structure (Cluster Splitting)

It is possible, that at some point, points from two distinct clusters could be placed in the same cluster. This case is illustrated in Figure 9. If two somewhat close clusters, where the closeness is relative to their natural variability, have points that are processed in an unfortunate order, they would go into a single cluster. The green points would create a covariance matrix that would drive themselves and all the future points from the two clusters into a single cluster. This would drive up the covariance estimate, and potentially result in other cluster's points being added to this cluster. This would impact detection performance, and any other action based on having separated clusters, including signature matching. This is also a very real possibility in the initialization step, where the first points collected in the image are automatically placed in the same cluster, but it is also likely to occur if the membership distance threshold is too large. Together, these facts necessitate the consideration of a step to potentially split clusters.

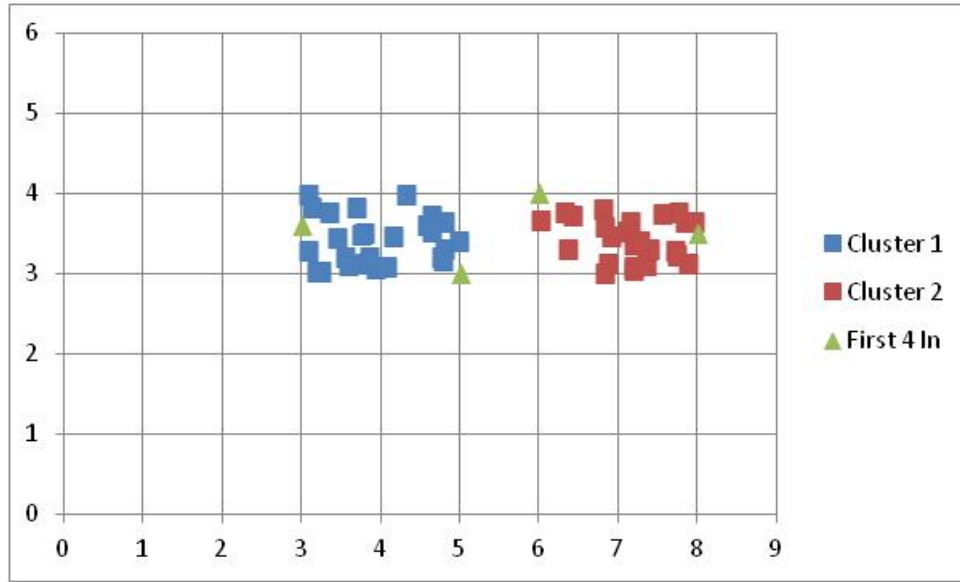


Figure 9: Potential Error Addressed by Splitting an Existing Cluster

When enabled, a check as to whether a cluster should be split is performed on each cluster to which a pixel is added, immediately following the update of the cluster when the pixel is added. This method was used to explore the behavior in the algorithm, but alternative methods of checking for splitting could result in substantial speed-ups of the algorithm. These are discussed in Chapter 5. To determine whether or not the cluster should be split into two clusters, the K-means algorithm is used, with k set to 2. As in X-means, once the cluster is split into two potential clusters, the delta BIC is calculated for the pair of clusters as described in Equation 5. Using the same rule as that for cluster merging, a positive value results in the cluster being split into two separate clusters, with updates performed in the distance tables, and membership lists.

As previously discussed in Chapter 2, the K-means algorithm is computationally expensive, with complexity $O(p_idkT)$. In contrast to the cluster merging step, which required no computation steps on the individual pixels previously assigned to the clusters,

other than potentially reassigning them in the membership lists, the splitting step, as implemented, requires the use of all pixels assigned to the cluster in the k-means algorithm. The number of pixels is expected to be much greater than the number of clusters that must be searched and updated in either the assignment or cluster merging step, which makes the cluster splitting the most computationally expensive step in the algorithm, when it is not disabled.

Anomaly Detection

A rudimentary anomaly detector was implemented for this algorithm. It is defined by three parameters which are the lag, the total memory, and the membership threshold used to declare the anomalies. The detector lags the ingestion of new pixels by a user selectable number of image lines, and compares the number of pixels in the cluster that contains the candidate pixel, to the number of pixels in a user selectable number of the most recently collected lines. If the number of items in the cluster with the candidate pixel is small relative to the total number of pixels in the recently collected lines, the pixel is declared an anomaly.

Figure 10 illustrates the behavior of the detector. The image illustrated has 5 pixels per line (lines are vertical) and is 8 lines long, for a total of 40 pixels. The pixels were read in the order listed starting with number one. Dark gray to the left of the figure indicates the pixels which are out of memory, and have been removed from the detector's counts. For this example, the memory is 4 lines. Light gray to the right of the figure are the pixels which have not yet been read. Pixel 33 has just been read. Pixel 13 has just been removed from the detector's counts. At this point, the green cluster contains 16 pixels while the blue cluster contains 3 and the red cluster 1. For this example, the lag is

set to 11, which means pixel 23 will be evaluated. Assuming the threshold of membership is greater than 1 or 5% of the 20 pixel memory window, it will be declared an anomaly. Whether or not pixel 22 was or was not declared an anomaly was determined prior to the reading of pixel 33 and the deletion of pixel 13 from the counts. Whether or not pixel 24 will or will not be declared an anomaly will occur after the reading of pixel 34 and the deletion of pixel 14 from the counts. Only the number of pixels in the cluster that contains a pixel which it is evaluated for anomaly declaration is used in making the anomaly declaration. Pixels 1 through 13 remain in their clusters, but they do not count toward the number of items in the clusters for anomaly declaration purposes.

1	6	11	16	21	26	31	36
2	7	12	17	22	27	32	37
3	8	13	18	23	28	33	38
4	9	14	19	24	29	34	39
5	10	15	20	25	30	35	40

Figure 10: Detector Behavior

The detector requires only the comparison of two numbers per pixel, and consequently has a $O(1)$ computational complexity. As implemented, the multiple thresholds and memory length combinations can be executed simultaneously as the algorithm runs, to allow for a more thorough and efficient evaluation of the detector behavior.

Algorithm Assessment Methodology

Three elements comprise the assessment of the algorithm. First, internal verification code will be used to ensure that the algorithm is implemented as intended. Second, a single image, chosen at random, will be used to explore the effect of changing the various parameters associated with the algorithm. Lastly, a set of selected parameters will be used to process a much larger set of images to validate the behavior seen in one image.

Algorithm Verification and Fault Detection Considerations

The algorithm, as implemented, includes a number of internal consistency checks to ensure correct implementation (or verification). These served to alert the developer to issues which may have occurred in the code. The checks are computationally intensive, but can be disabled by use of the appropriate flags.

The first check is against the combination formulas for the combined covariance matrices. The results of these calculations are compared against the covariance values computed directly on the pixel values. If all elements of the covariance matrices match to within a value of 0.1 (covariance values for this data set easily surpassed several thousand), the calculation was considered correct. This check against the values also verified the mean and individual covariance update formulas used to derive the covariance matrices used in the formulas. Also, the determinants of covariance matrices are checked to ensure that they are positive, and all delta BIC calculations are checked for imaginary components (a result in the BIC formula of having a negative determinate of a covariance matrix), which would indicate an error in calculation.

At the end of the algorithm, the list of pixels in each cluster is compared against the cluster identity stored for each pixel. Any mismatches are reported. Further, the total number of pixels in the cluster membership lists is compared against the total number of pixels processed to ensure that the two numbers are equal.

As the algorithm runs, an optional “heartbeat” display is selectable to allow output to the screen at 1000 pixel increments, listing the number of clusters and time elapsed during the processing of the previous 1000 pixels.

Tuning

The Mahalanobis distance threshold, the number of principal components, and the value for lambda (if used to tune the delta BIC calculation) must be set for the clustering algorithm. The anomaly detection portion, which relies on the clustering performance, additionally requires that the length of the pixel memory and the cluster size thresholds be determined as well. A single image, randomly selected from the available images will be used to determine operating settings, prior to running the algorithm on the remaining images to assess performance. As no ground truth is available for use in determining the accuracy of the clusters themselves, the anomaly detection accuracy of the rudimentary cluster based detector will be used to inform the determination of the parameters in addition to the results of the preliminary experiments described at the beginning of this chapter. Clustering will also be assessed visually. Speed is a concern, the time required for the algorithm to run will also be measured and considered in determining the operating parameters.

At each combination of Mahalanobis distance threshold, number of principal components, and lambda settings, numerous detectors can be assessed (without impacting

any other aspect of clustering performance). Essentially, this creates a split plot experimental design condition, with the easily alterable detector parameters being at the split plot level and clustering performance at the whole plot level. The Mahalanobis distance threshold is specified in terms of the square of the Mahalanobis distance.

Adequate computing resources were available to support mesh grid experimental design in the three clustering parameters and two detector parameters. Design parameters are shown in Table 2. Initially, the use of merge cluster and split cluster in the improve structure step of the algorithm were considered as additional factors in the design, but for reasons to be explained in the results section, they were quickly removed from consideration. Were additional images used in the initial experiments, a random effects or robust parameter design (RPD) experiment would have resulted. As no available weights are available for variance verses performance, an RPD was not considered, but could be an area for research in the future.

Table 2: Experimental Design Factors and Levels

Factor Name	Levels
Mahalanbois Distance (squared) Threshold	50 to 2000 in increments of 50
Principal Components Retained	2 to 43 (in 1 principal component increments)
Lambda	1, 10, 15, 20, 25, 30, and 35
Pixel Memory	1 to 10 lines (1 line increments)
Anomaly Detection Threshold	2% to 20% of the number of pixels defined by the Pixel Memory (2% increments)

For detector performance, the anomaly detector can only assess whether a given pixel is or is not an anomaly. In each case, it can be either correct or incorrect. This results in the four outcomes shown in Table 3.

Table 3: Detector Outcome

	Actual Anomaly	NOT an Actual Anomaly
Anomaly Declared	True Positive (TP)	False Positive (FP)
Anomaly NOT Declared	False Negative (FN)	True Negative (TN)

The four outcomes are often expressed as rates, such as a true positive rate or a false negative rate. The TP and FP rates will be used as defined in Equation 9 and Equation 10 respectively. Alternatively, the FN and TN rates could be used, as they are each the 1.0-TP rate and 1.0-FP rate respectively. All rate calculations will be made on a pixel by pixel rather than a target by target basis.

$$TPr = \frac{TP}{TP + FN} = \frac{TP}{P_{tot}} \quad \text{Equation 9} \quad (9)$$

Where:

TP = Count of True Positives

FN = Count False Positive

P_{tot} =Count of total positives

$$FPr = \frac{FP}{TN + FP} = \frac{FP}{F_{tot}} \quad \text{Equation 10} \quad (10)$$

Where:

TP = Count of True Positives

FN = Count False Positive

F_{tot} =Count of total negatives

Obviously, a higher true positive and lower false positive rates are preferred. For given applications, one of the rates might be substantially more important than the other. For the purposes of evaluating this algorithm, neither is considered more important than

the other, so an equal weight may be tied to each. While methods exist to deal with these two separate values as individual response variables when selecting the operating parameters, they can be combined into a single quantity by way of a Receiver Operating Characteristic (ROC) curve, which plots the true positive rate against the false positive rate, giving an indication as to the available performance selections for a detector. The area under the ROC curve (AUC) can be used to compare various detectors when equal costs are associated with the various possible errors. ROC curves begin in the (0,0) point, where nothing is ever declared an anomaly, and follow some path to reach the (1,1) point, where everything is declared an anomaly. Desirable ROC curves are those that remain as far from the line connecting (0,0) to (1,1) as possible. This line, known as the chance line, indicates the performance expected from random guessing. For many detectors, a user selectable parameter can shift the detector along the ROC curve, trading off one rate for the other. The individual detectors to be considered in this algorithm do not possess this feature, and generate a single true positive rate and single false positive rate number per run. This precludes drawing a full ROC curve for each detector. An approximation, using the single point connected to the (0,0) and (1,1) points of the ROC plot is considered a suitable surrogate to the full ROC curve for assessing the clustering algorithm and detectors performance.

For the single point ROC curve area calculation, the AUC can be calculated by assuming the selected point is a corner of two triangles, the sum of whose area equals the AUC. One triangle formed with the points (0,0) and (1,0), and the other formed with the points (1,0) and (1,1). In each case, the base of the triangle is unit length, and the height

is defined by the distance of the point to the appropriate axis. The resulting formula for AUC is shown in Equation 11.

$$AUC = \frac{1}{2}(TPr + (1 - FPr))$$

Equation
11
(11)

Where:

TPr = True Positive Rate

FPr = False Positive Rate

As can be seen in Figure 11, a single AUC may be due to several possible combinations of TPF and FPF. An interesting feature of this calculation method is that a given AUC defines a locus of TP and FP rates, which fall on a line. This is illustrated in Figure 11, by three separate points corresponding to an AUC of .8, which all fall on a straight line connecting points (0,.6) to (.4, 1).

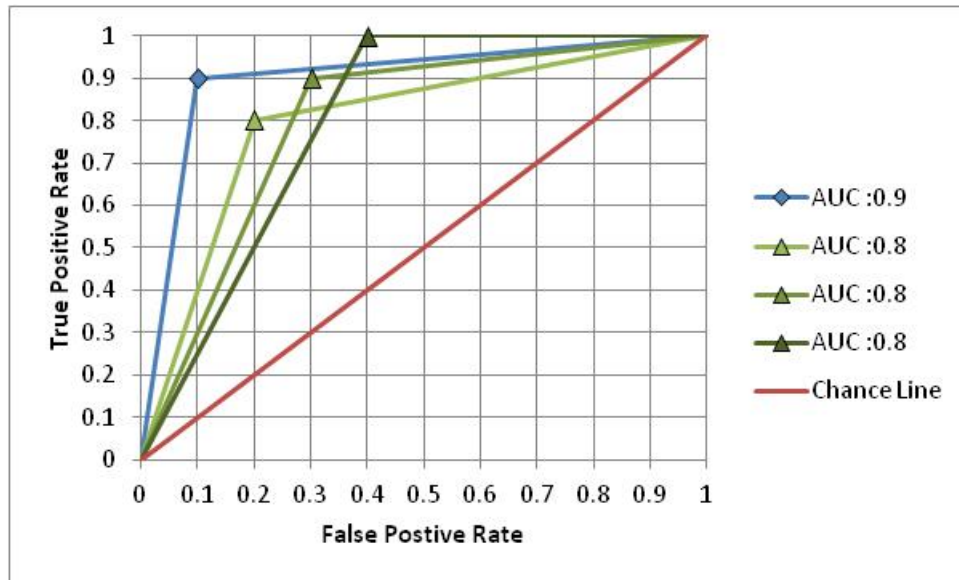


Figure 11: Area Under the ROC Curve For Single Points

In addition to detector performance, several additional measures will be collected to provide greater insight to the operation of the algorithm. A list of those parameters is shown in Table 4.

Table 4: Response Variables

Factor Name	Description
Total Clusters	Total number of clusters in the image at the conclusion of the algorithm
CI70	Minimum number of clusters required to contain 70% of the pixels in the image
CI80	Minimum number of clusters required to contain 80% of the pixels in the image
CI90	Minimum number of clusters required to contain 90% of the pixels in the image
CI95	Minimum number of clusters required to contain 95% of the pixels in the image
CI96	Minimum number of clusters required to contain 96% of the pixels in the image
CI97	Minimum number of clusters required to contain 97% of the pixels in the image
CI98	Minimum number of clusters required to contain 98% of the pixels in the image
CI99	Minimum number of clusters required to contain 99% of the pixels in the image
TPR	True Positive Rate
FPF	False Positive Rate
TNR	True Negative Rate
FNR	False Negative Rate
AUC	Area under the ROC curve
Count of Combinations	Number of times clusters were merged
Count of Euclidean/Mahalanobis Mismatches	Number of times a pixel was assigned to a cluster to which it was not the closest from a Euclidean perspective
Time in Assign Step	Total amount of time spent assigning pixels to clusters during the processing of the image
Time in Combination Step	Total amount of time spent combining or determining whether to combine two clusters during the processing of the image
Total time	Total amount of time required to process the image (in seconds)

Performance Validation

After the conclusion of experimentation on a single image, parameters will be selected for use on 17 total images (including the image used for parameter development). A subset of the parameters from Table 4 will be reported for each image.

Summary

This chapter has outlined a battery of assessments to determine the behavior of the common HSI pixels in the principal component subspace, and the behavior of the Mahalanobis distance in that subspace. If desirable behavior is exhibited in the subspace, the algorithm outlined herein should be capable of performing online clustering of HSI images and should simultaneously be able to support an online detector. A second set of evaluations was presented which can characterize the algorithms performance in terms of the user defined parameters, determine a set of operating parameters, and ultimately demonstrate the clustering behavior of the algorithm over a set of images.

IV. Analysis and Results

Chapter Overview

This chapter opens with the results of several experiments which characterize the behavior of hyperspectral imagery pixels in the principal component subspace used by the online clustering algorithms. The first few experiments are designed to determine whether Euclidean distances can be effectively used in the online clustering algorithm in the principal component subspace, and give an indication that Euclidean distances are inadequate. Additional experimentation focuses on the impact of various common background pixel types of the structure of the principal component subspace. A final set of preliminary experiments further confirm the adequacy of Mahalanobis distance for this algorithm, and lead to a suggested dimensionality and distance threshold for use in the algorithm.

Following the results from the preliminary experimentation, the results from an exhaustive sweep of the three clustering parameters and two detector parameters performed on a single randomly selected image are presented, characterizing the algorithms performance. Lastly, a presentation of the results of the algorithm run on over a dozen representative hyperspectral images is provided.

Preliminary Confirmation of Insufficiency of Euclidean Distance for Clustering

The results of the two experiments run to determine whether it might be possible to use Euclidean rather than Mahalanobis distances in a hyperspectral clustering algorithm appear below. The results in both cases point to Euclidean distances being inadequate.

Condition Number Test Results

For the first experiment, the (Mathworks, 2012) k-means algorithm with squared Euclidean distances was run on a sample hyperspectral image using various combinations of dimensions (principal components) and directed numbers of clusters. The dimensions ranged from 2 to 43, and the directed number of clusters ranged from 2 to 51. For each cluster, a covariance matrix was constructed and the condition number, defined as the ratio of the largest to the smallest Eigen value, was calculated. For each combination of number of clusters and dimensions, the base 10 logarithm of the maximum and median and minimum condition numbers are displayed in Figure 12 ,Figure 13 and Figure 14 respectively. A logarithmic scale was required due to both the range of values, and the presence of several extreme values in all three data sets. When the number of principal components was small (low single digits), the condition numbers can be seen to remain less than 10^3 in the extreme and less than 10^1 in at least 50% of the clusters (by definition of the median). While a few extreme cases show at least one cluster with a condition number that would produce numerical instabilities (greater than 10^{17}), the potential numerical stability issues are not the cause of greatest concern. Inherently, the k-means algorithm assumes a single variance estimate in all directions and consequently assumes spherical clusters. The condition number, as defined earlier, provides a measure of the ratio of the largest axis to the smallest axis (in terms of variance) of the cluster. For k-means to be appropriate, a value near one would be desirable. As experience has shown numerous detectors require consideration of a dozen or more principal components of a hyperspectral image for desirable operation, and the distribution of condition numbers,

regardless of cluster number, in this region is not close to one, Euclidean distances should be considered suspect.

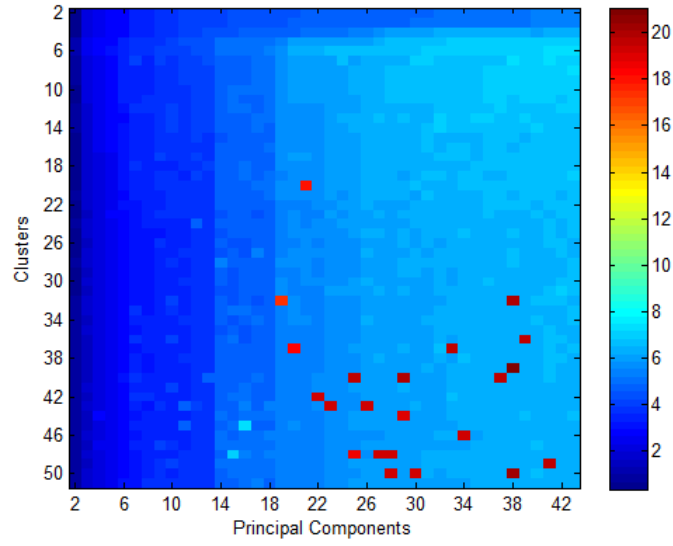


Figure 12: Maximum K-means Covariance Condition Numbers (log₁₀ scale)

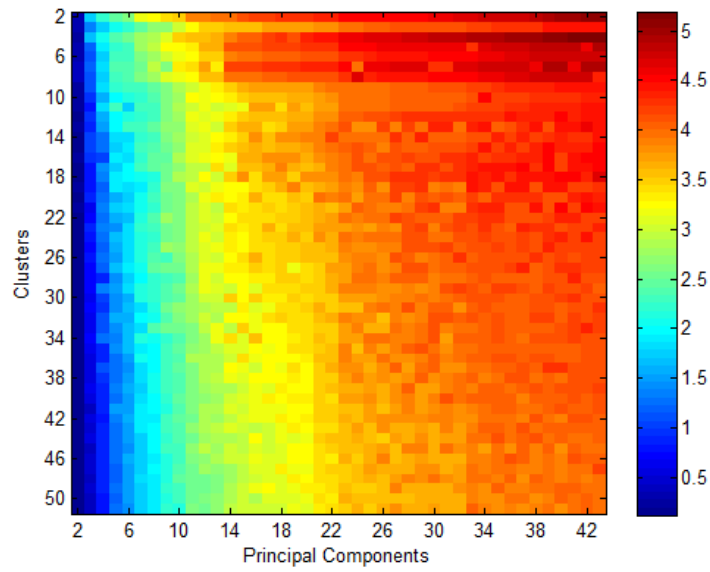


Figure 13: Median K-means Covariance Condition Numbers (log₁₀ scale)

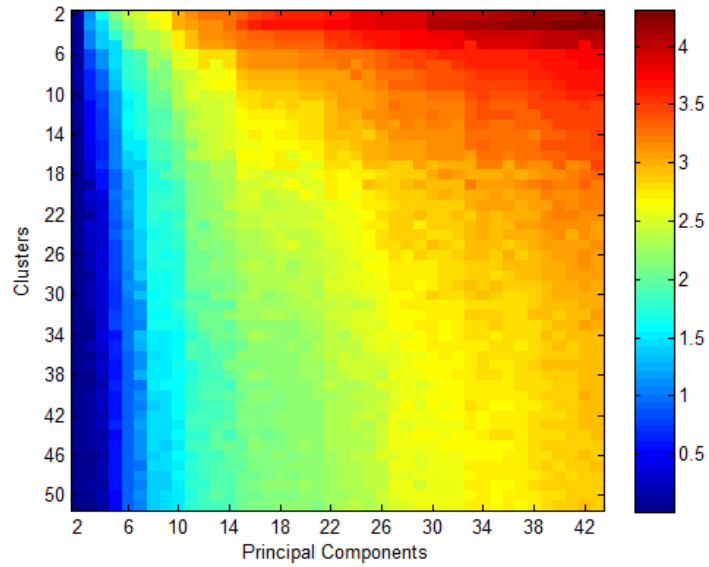


Figure 14: Minimum K-means Covariance Condition Numbers (\log_{10} scale)

Clustering Test

The second experiment involved running the same k-means algorithm using Euclidean distances for a small sample of cluster sizes and two principal components from a typical hyperspectral image. The selected image is shown in natural colors in Figure 15.

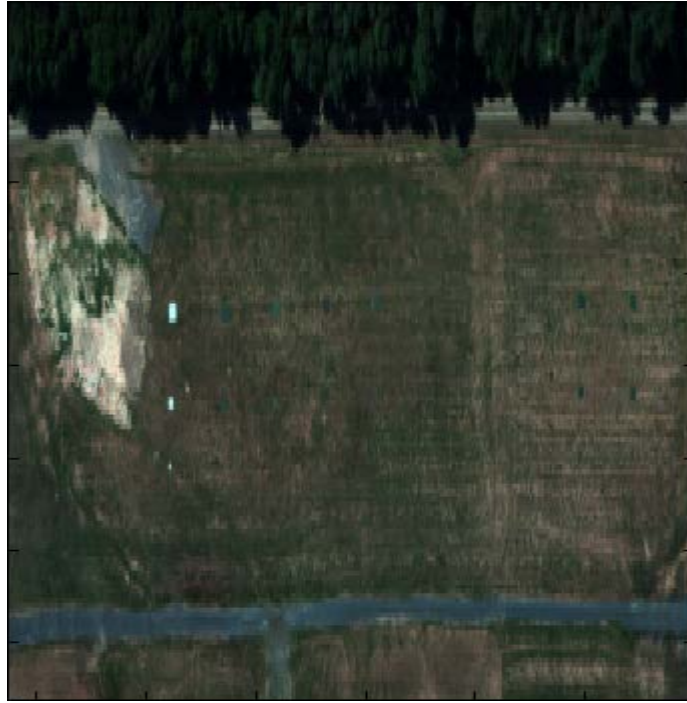


Figure 15: Randomly Selected HSI Image

Figure 16 shows the first and second principal component scores for each pixel in the randomly selected image. Each pixel is displayed in the natural colors used in Figure 15. The most striking and immediately apparent observation from the figure is the clear separation into what appears to be three clusters. The use of natural colors allows the viewer to identify the streaking dark area to the upper left as the tree pixels from the image. The bluish color area in the lower left can be identified as the road pixels, and the green to brown cluster in the middle of the image appears to correspond directly to the open field area in the image. The use of natural color does not appear to have seen popular use in the literature, but this example provides a clear indication of its utility.

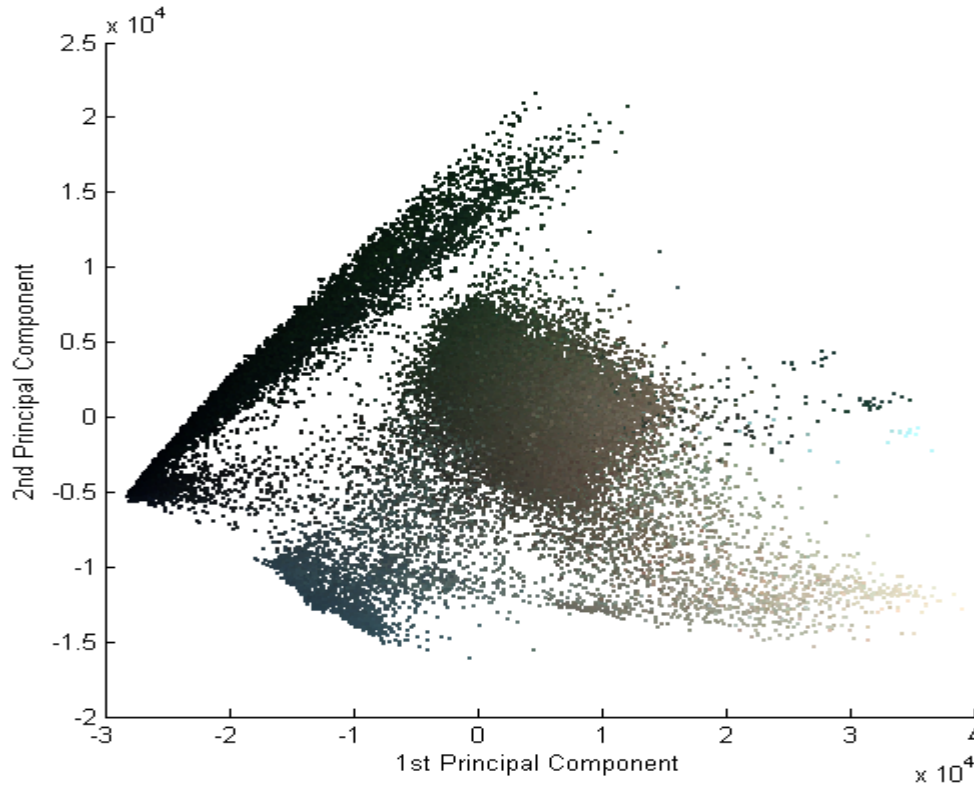


Figure 16: Hyperspectral Image Pixels Plotted in the First Two Principal Components

Prior to performing k-means clustering on this data set, the anecdotal evidence shown in Figure 16 appears to show that the HSI domain with reduced dimension using PCA methods indeed requires a non-Euclidean distance measure. The tree cluster would not be well approximated by a round cluster.

With three signatures appearing to cluster together in Figure 16, running k-means with $k=3$ would seem to be a natural choice. The result of this clustering is shown in Figure 17, with each cluster being indicated by a unique color. The resultant clustering does not appear to separate the clustering in the same way a human might. Clusters are formed across evident gaps between sets of points, which was listed earlier as an

undesirable clustering behavior. Contiguous sets of points are also arbitrarily split by clusters, which is another undesirable behavior.

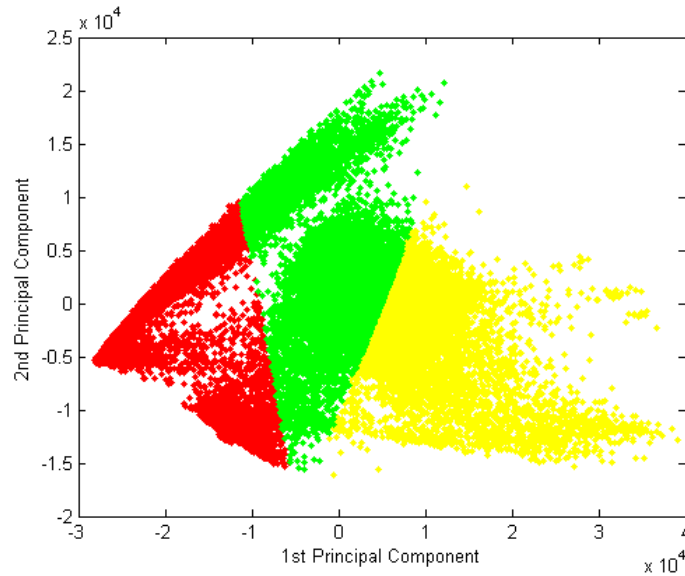


Figure 17: Clustering in 2 PC's using K-means (k=3) & Euclidean Distances

Setting $k=5$ and running k-means again results in the clustering shown in Figure 18. The cluster shown in red appears to capture the road pixels uniquely, but the lumping together of some field with some tree pixels in the same cluster is not desirable. Also, the tree cluster is again split at an arbitrary location into two clusters.

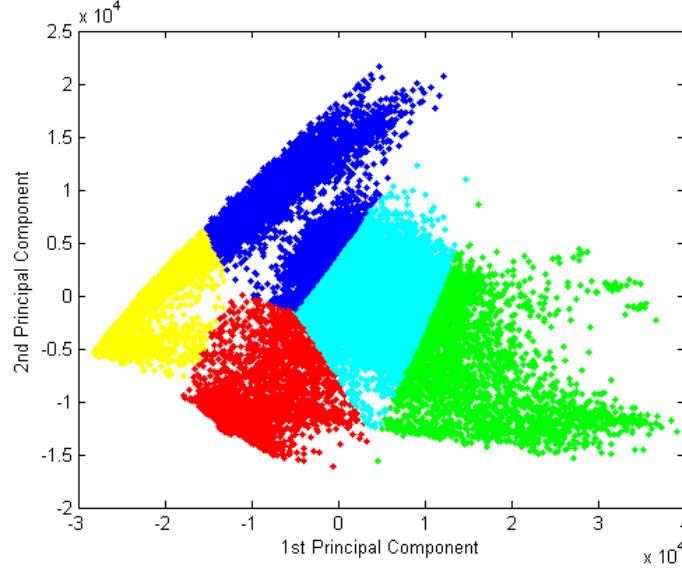


Figure 18: Clustering in 2 PC's using K-means (k=5) & Euclidean Distances

Ideally, the cluster boundaries would occur in the low density (or empty) areas clearly visible to the human eye. Contiguous regions would not be split into different clusters, unless a substantial decrease in density at the boundary of the two regions would dictate it. The clustering results observed in Figure 17 and Figure 18 do not exhibit these desirable characteristics. In fact, with regard to the dark blue cluster in Figure 18, the undesirable pattern illustrated in Figure 6 is pronounced. Available evidence appears to support the use of Mahalanobis distance over Euclidean distance in the clustering algorithm, despite the added computation cost.

It is interesting to note from these plots that the major components of the background appear to be split out in the first two principal components. A few of the outlying pixels in the right side of the image are known to be anomalies, but a majority of the anomalies are located in the middle of the field pixels in Figure 16.

Characterization of the Principal Component Subspace and Mahalanobis Distance Behavior

To characterize the behavior of expected clusters in the principal component subspace, random patches of pixels, typical of the backgrounds in the available images, were extracted and placed into various combinations of data sets. Specifically, pixels for trees, dirt, and road were taken from a forested image, and sand pixels were selected from a desert scene. A random selection of other, anomalous pixels was also extracted. Figure 19 shows typical natural spectra in the retained bands of these pixel selections. Band to band correlation is evident, which provides some of the motivation for using principal components.

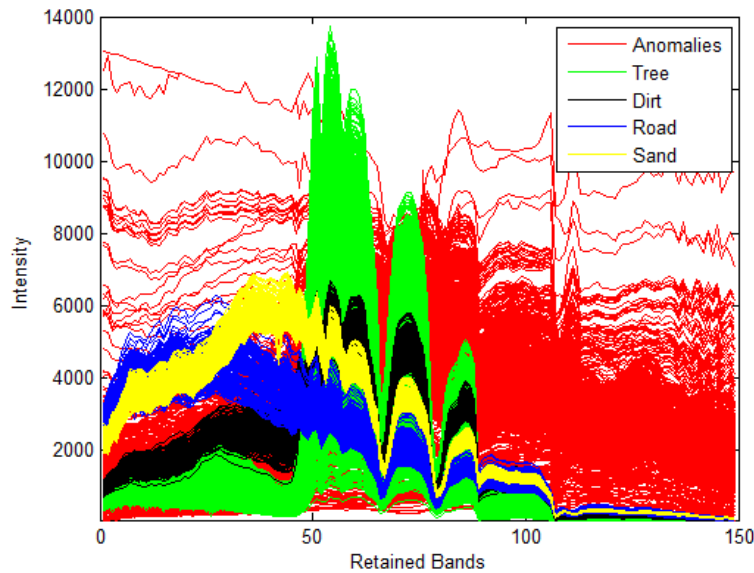


Figure 19: Spectrum of Various Pixel Types

Eigen Values by Signature Test Results

A plot of the \log_{10} of the Eigen values of the covariance matrices for the trees, dirt, road, and the second type of sand all taken separately are shown in Figure 20. The Eigen value gives a relative measure of the amount of total variation contained in the

given principal component. The \log_{10} scale was used over natural units due to the extraordinary range of values present in the first few principal components. The similar pattern for each background set of pixels gives an indication that the amount of variation contained in the principal components is similarly distributed regardless of specific background, and that the dimensionality of the information contained in the data does not substantially vary by background. The anomalous pixels contain a substantially greater overall variation in each principal component beyond the first. Taken individually, the dimensionality appears to be defined by a local “leveling out” of the slope which occurs around 10 to 15 principal components for the background pixels, and at a higher value for the anomalous pixels. This leveling is normally assessed in a natural rather than a log scale, but this assessment is considered sufficient based on further results to be presented shortly. Were the y axis in natural rather than \log_{10} units, only the first one or two principal components likely would have been retained by visual inspection due to their dominance of the scale settings.

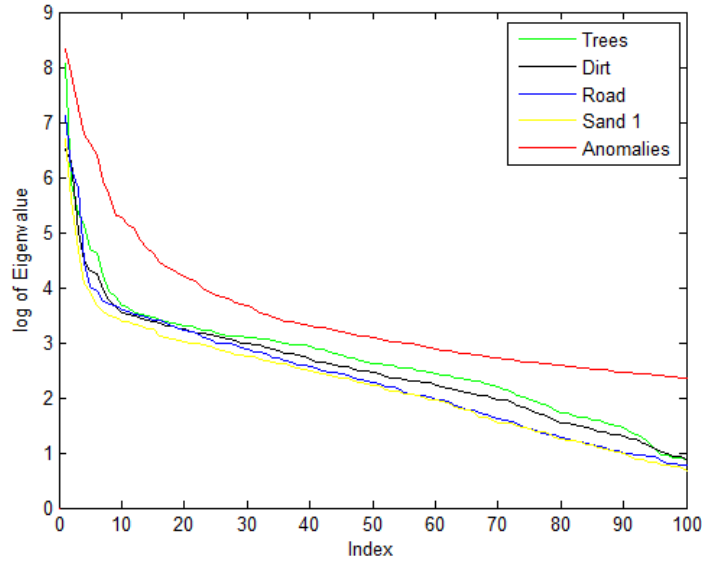


Figure 20: Eigen values of Selected Pixel Covariance Matrices

Eigen Values by Combined Signature Test Results

A similar plot, comparing the Eigen values of the tree, grass, and dirt pixels with and without sand and road pixels are shown in Figure 21. All background signatures combined with an additional random collection of anomalies is also shown. The anomalies accounted for 1% or 10% of the total number of pixels as shown. Without the anomalies, the dimensionality appears to be independent of the exact combination of backgrounds, and appears to match the variance over principal component distribution of any single background in Figure 20. The additional of anomalies alters this pattern by inflating the variance in the region from 5 to 20 principal components. It appears that the greater the percentage of anomalies, the greater a disturbance to the trend.

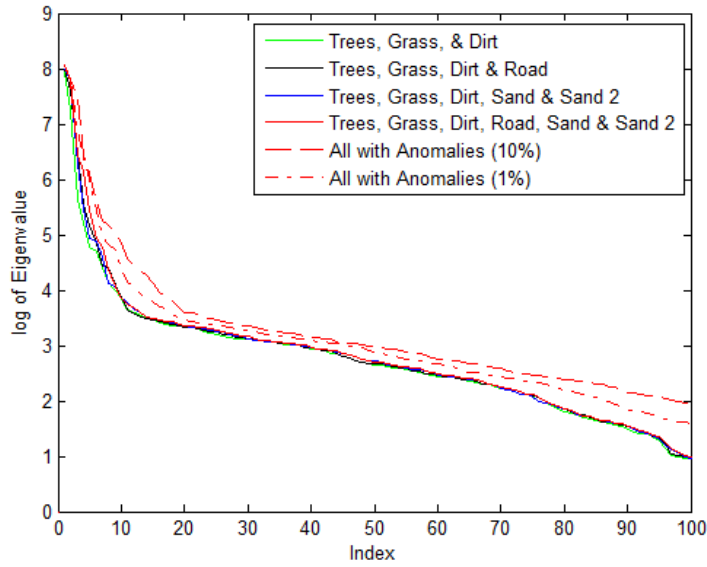


Figure 21: Eigen values of Selected Combined Pixel Covariance Matrices

While the distribution of variation among principal components appears to be independent of actual background (for collections of background pixels), this does not mean that the actual principal component loadings (which capture how the individual principal components are constructed from the full data set), do not vary based on background.

Loadings of Paired Signatures Results

To determine if possible differences in background could drive different loadings, the collection of tree and road pixels were placed into a single data set, and the principal component loadings for the first principal component were calculated. This was also accomplished for the dirt and road pixels together. The loadings of the first principal component in each case are shown in Figure 22.

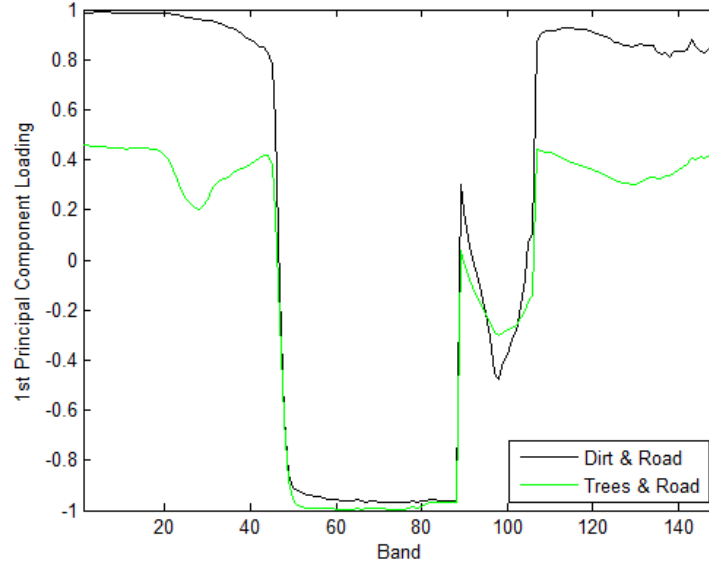


Figure 22: 1st Principal Component Loading Plot

While an overarching pattern exists, the actual loadings differ, indicating that the actual background will influence the linear transform used when constructing the various principal components. Figure 23 shows the mean difference in each band between the road and tree pixels divided by the standard deviation of the values in each band. The same information is shown for the dirt and road pixels. The similarity of Figure 23 to Figure 22 provides an indication that the first principal component is capturing the variation explained by the differences in the two backgrounds in each case, which implies the principal component construction is dependent on the image's background content. In the worst case, this could mean a very different subspace exists for every new image, and that this subspace is only calculable after the collection of the entire image. That could preclude the use of the principal components for use as the appropriate clustering sub-space. This would, however, only be the case if the distance measure used by the algorithm were to be affected by the differences in subspaces.

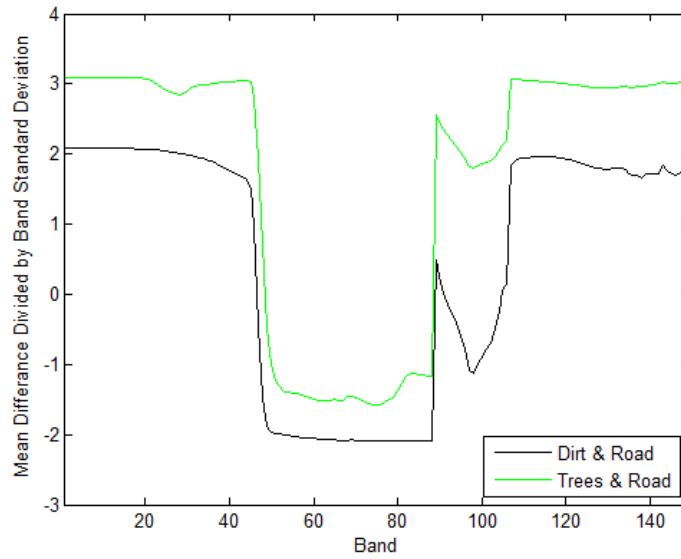


Figure 23: Mean (by band) divided by standard deviation

Distance Plots Results

To determine the applicable behaviors of the Mahalanobis distances in the principal component subspace, the Mahalanobis distances from various pixels to both a road and a tree cluster (formed from the road and tree pixels) were calculated for subspaces built from 1 to 40 principal components. The results for the road and tree clusters are shown in Figure 24 and Figure 25 respectively. Similar plots using only sand, or combinations of sand and dirt yielded similar results. For the pixels selected from common image backgrounds, a Mahalanobis distance cutoff value between 10 and 15 from a cluster center captures elements in the cluster and excludes items outside of the cluster for dimensions above approximately 15. It also appears that above approximately 10 principal components, the contribution of each additional component to the distance measurement is relatively minor. This supports a dimensionality assessment of

approximately 15, which matches previously assessed dimensionalities in anomaly detectors.

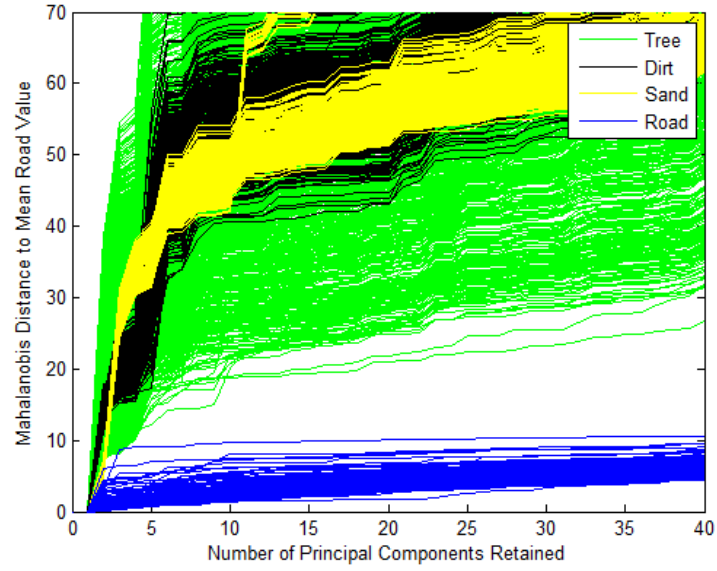


Figure 24: Mahalanobis Distance of Selected Pixels to Mean Road Value

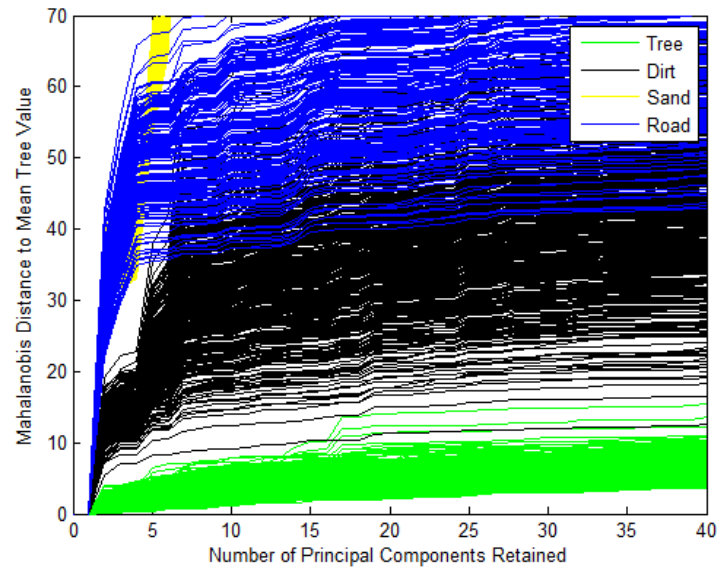


Figure 25: Mahalanobis Distance of Selected Pixels to Mean Tree Value

To verify that the addition of anomalies would not alter this apparent property of the subspace adversely, the same calculations were performed with anomalous pixels accounting for 5% of the total number of pixels when the principal components were calculated. The results are shown in Figure 26 and Figure 27. The observed pattern still holds.

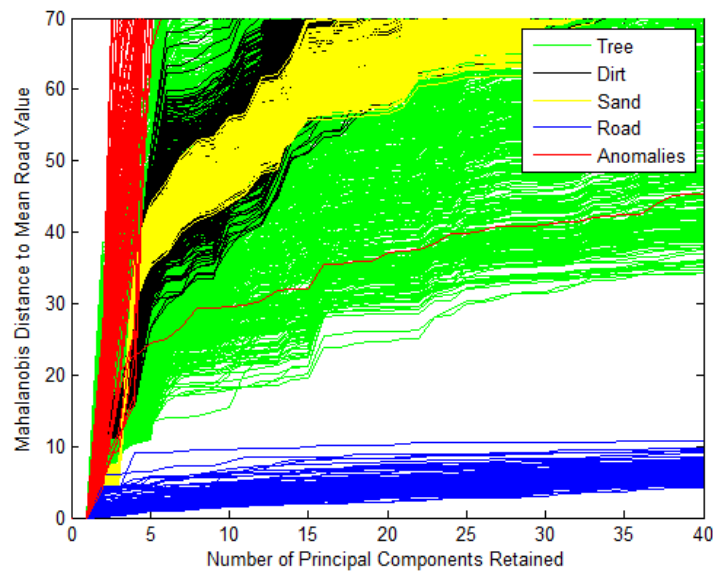


Figure 26: Mahalanobis Distance of Selected Pixels to Mean Road Value (with Anomalies)

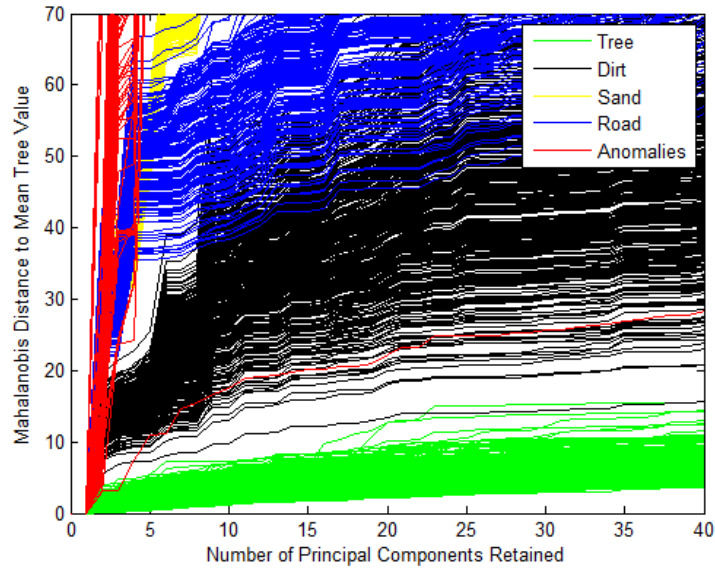


Figure 27: Mahalanobis Distance of Selected Pixels to Mean Tree Value (with Anomalies)

To again verify the inadequacy of Euclidean distances, the Euclidean distances from between various pixels and the mean of the tree cluster were calculated and are shown in Figure 28. No clear threshold exists, and any threshold that would capture all of the tree pixels would capture the entire dirt cluster, which is undesirable. Mahalanobis distances are still clearly preferred. It is interesting to note that the Euclidean distances do not noticeably change beyond 5 principal components.

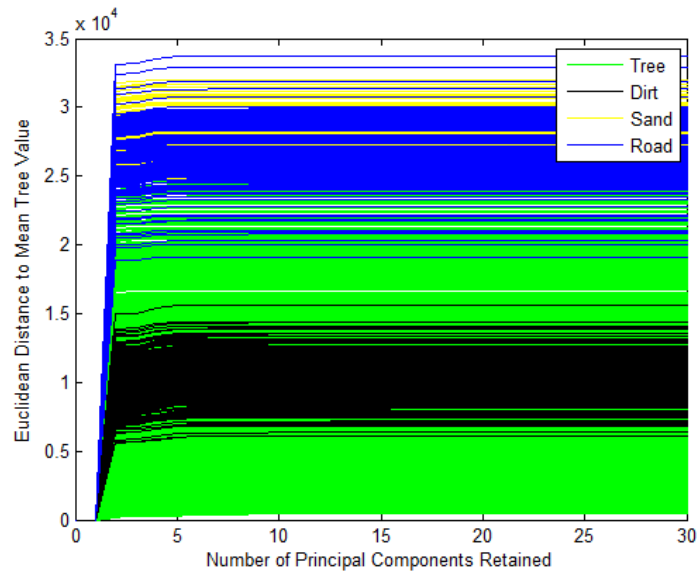


Figure 28: Euclidean Distance of Selected Pixels to Mean Tree Value

The behavior, where a single Euclidian distance is somehow captures another cluster completely without fully capturing the intended cluster is easily explained by referencing Figure 29. The tree cluster in green is an eccentric ellipse with points in the cluster farther away from the center than the center of the dirt cluster (or parts of the road) clusters. This behavior is clearly developed in the first two principal components likely due to a majority of the variation being contained therein.

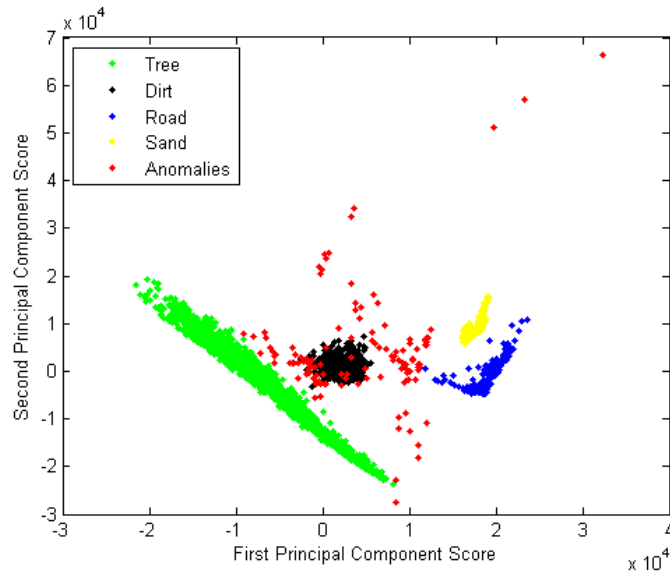


Figure 29: Selected Pixels plotted in the first two principal components

The anomalies shown in Figure 29 would potentially appear as members of the existing clusters, or random outliers, but are clearly separate in higher dimensions as can be seen in a plot of the principal component scores. Figure 30 shows all scores for all principal components of the same pixels shown in Figure 29. The range of scores in the higher components tend to zero, relative to the first few components.

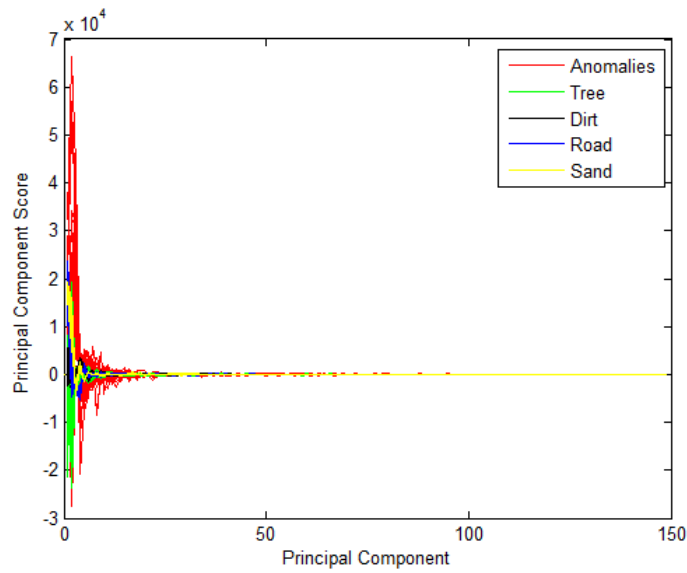


Figure 30: Principal Component Spectrum of Various Pixel Types

Figure 31 shows a view of the same plot with emphasis on the first 20 principal components. Although no “in between” principal components exist, each pixel is a plotted with a line connecting its adjacent principal component scores. This allows patterns to be visible to the human eye that might otherwise be lost if the individual scores were not connected together. The anomalies show no particular pattern in their progression from the first to the second principal component, unlike the road, sand, and dirt pixels, which can be seen to have a fairly tight clustering behavior in Figure 29 which is nearly discernible with reference to Figure 31.

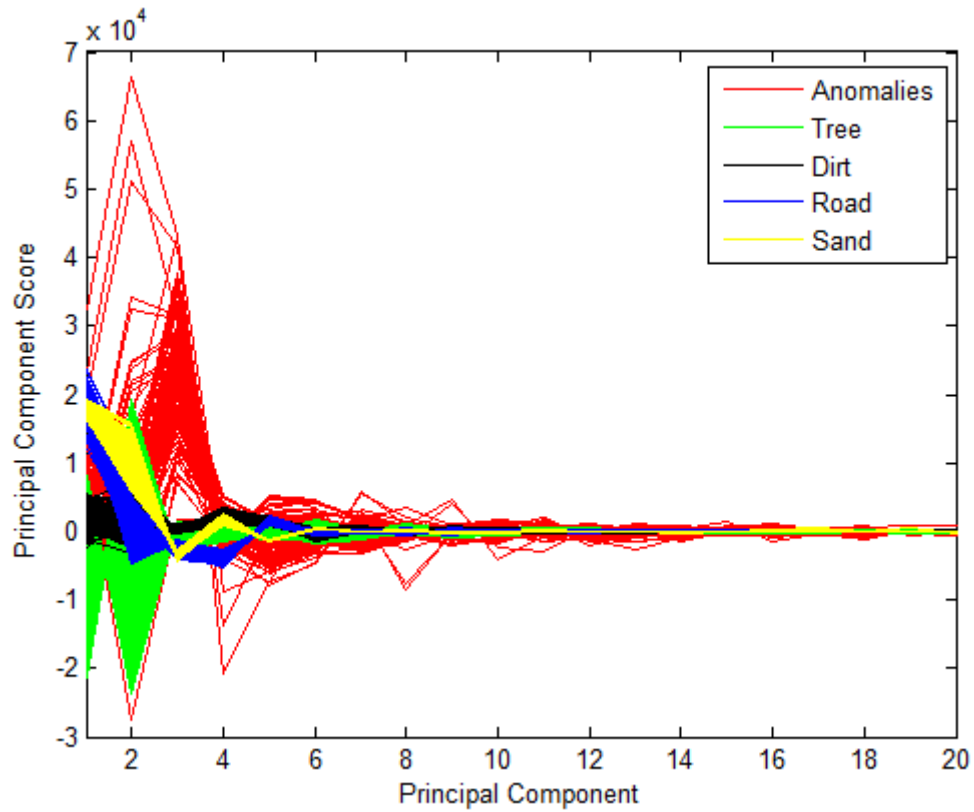


Figure 31: Principal Component Spectrum of Various Pixel Types (1-20)

It would appear from this plot that the third principal component alone may contain the information required to separate the anomalies from the other signatures. A closer version view of this region is shown in Figure 32. This trend becomes even more evident when the third principal component is plotted against the first in Figure 33.

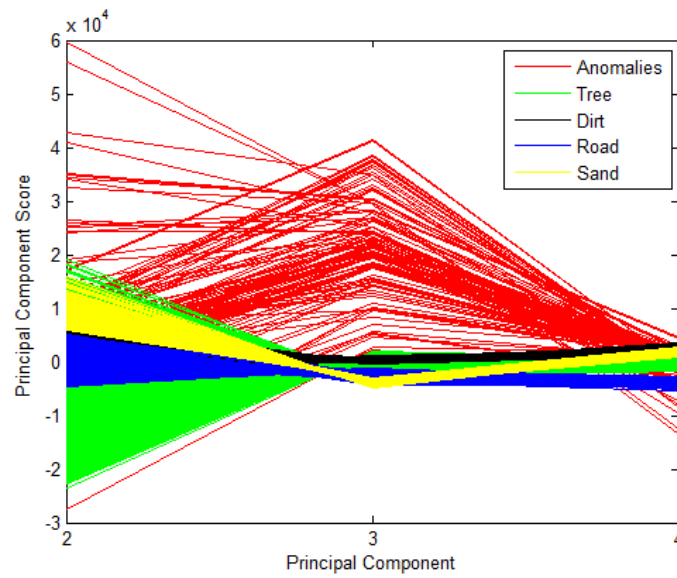


Figure 32: Principal Component Spectrum of Various Pixel Types (2-4)

This trend can also be seen when the third principal component is plotted against the first in Figure 33.

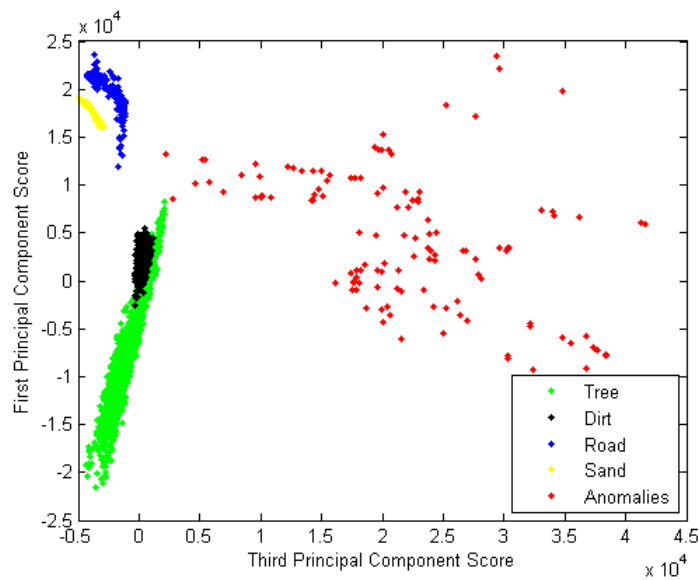


Figure 33: Principal Component Plot (1st versus 3rd)

The tendency of the anomalies to have relatively high scoring behavior across various principal components other than the first few can be seen in further detail in Figure 34, which focuses on a midrange section of the principal components.

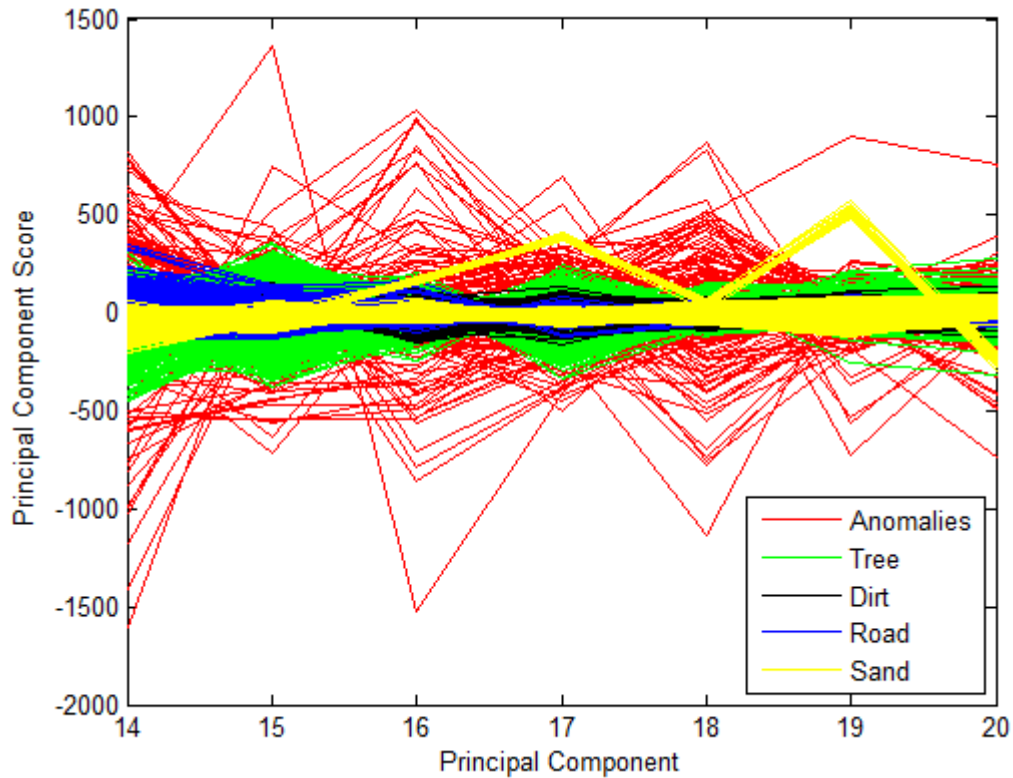


Figure 34: Principal Component Spectrum of Various Pixel Types (14-20)

The behavior of the anomalies in the principal component sub-space suggests that a detector could potentially be developed by implementing an ensemble of individual detectors examining anomalous behaviors in each separate principal component using the methods described by (Turnquist, 2011). The ensemble behavior could capture the trend that when a given anomaly appears to score far from the x-axis in a single component, it does not do so in the adjacent component, and some background pixels may score similarly high in a few of the components, but not a majority. Further, the ability to

identify which principal components may separate anomalies, various backgrounds, or other behaviors in this type of plot suggests image processing tool using a display to assist in identifying unusual behaviors in an image. For a random image, ground truth would not be available but a plot done in natural colors or other selectable false colors such as the one show in Figure 35, could still be very helpful.

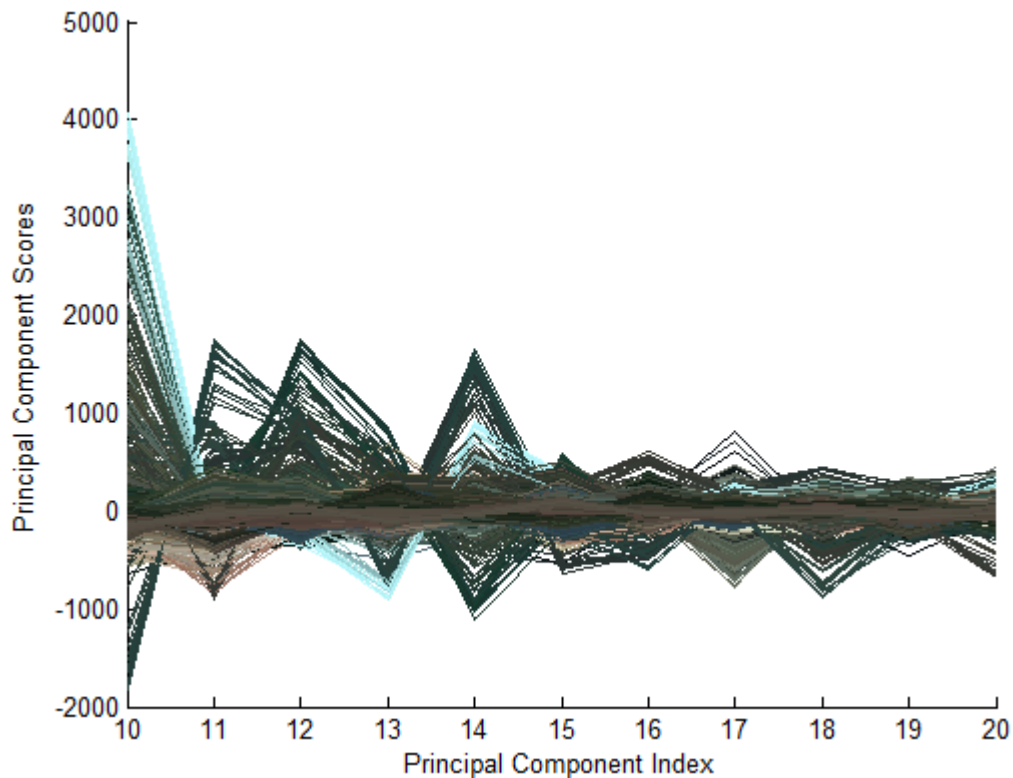


Figure 35: Pixels Plotted in Natural Color in Selected Principal Components

Ideally, individual blocks of pixels such as those at the “peaks” in a given principal component on the display could be selected and highlighted on the original image, which itself could be displayed in an arbitrary band or combination of false or true colors. An example of what this tool might display is shown in Figure 35, the presumably user controlled blue block has been used to highlight the peaks in principal component number 14. These are highlighted in the principal component display and the

corresponding pixels in the image are known anomalies. This could be remarkably helpful to an analyst charged with processing. Both the development of an ensemble detector based on principal component outliers and an image processing tool based in principal component space are areas suggested for future work in Chapter 5.

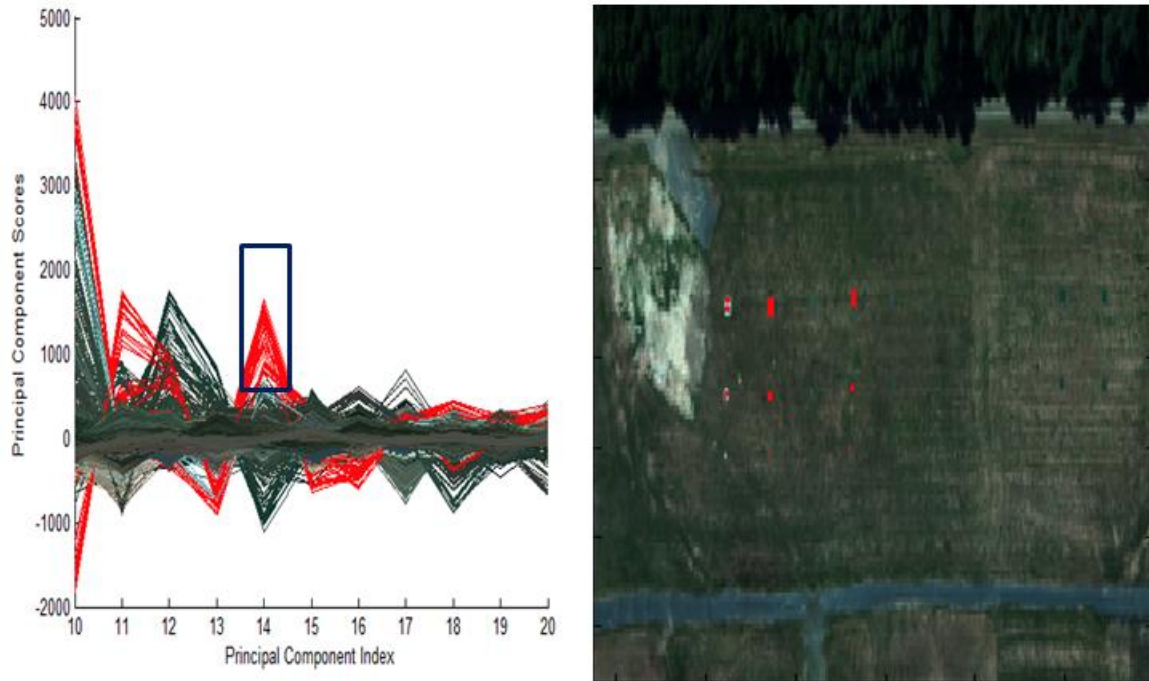


Figure 36: Potential Analysis Tool GUI Example

Despite the interesting behavior of the anomalies in the higher dimensions of the principal component space, background pixel behavior appears to be well established in the first few principal components. It may even be the case that background centric clustering could be accomplished in a lower number of dimensions than anomaly detection, if anomalies need not be separated from the background by the clustering algorithm. In any event, the behavior of the Mahalanobis distance appears stable with regard to potential common background composition. In fact, a dimensionality of 15 and

a Mahalanobis distance threshold of 15 for cluster membership might be sufficient for clustering for images collected by this sensor.

Tuning Procedures on a Single Image

The same randomly selected image used previously for the preliminary experiments was again used for the more extensive parameter experiments. The verification flags built into the code were used initially until all implementation errors were located and corrected. Following verification, preliminary experiments using the four Euclidean distance rules for clusters without covariance matrices resulted in a degenerate behavior for rules 2 and 3. Specifically, they generated an unacceptably high number of clusters in the several thousands regardless of parameter settings. Rule four never resulted in the assignment of a point to a new cluster by itself, indicating that rule 1 would have identical results with less overhead. The rule 1, where a pixel is assigned to the nearest cluster from a Euclidean distance perspective if that cluster does not have enough points to generate a covariance matrix, was used for all further experimentation. Unexpectedly, an experiment where all subsequent pixels, regardless of measure, were assigned to the newly formed cluster until a covariance matrix was generated produced results similar to those reported below.

Figure 37 shows the number of clusters produced by the algorithm at the conclusion of the processing of the image. For this run, the lambda parameter for the delta BIC was set to 1, which corresponds to the traditional definition of the BIC. A \log_{10} scale is used due to the range of values seen in the response. As would be expected, a large Mahalanobis distances and small numbers of principal components results in all

pixels are assigned to the same cluster. For small Mahalanobis distances, an excessively large number of clusters are generated depending on the number of principal components retained.

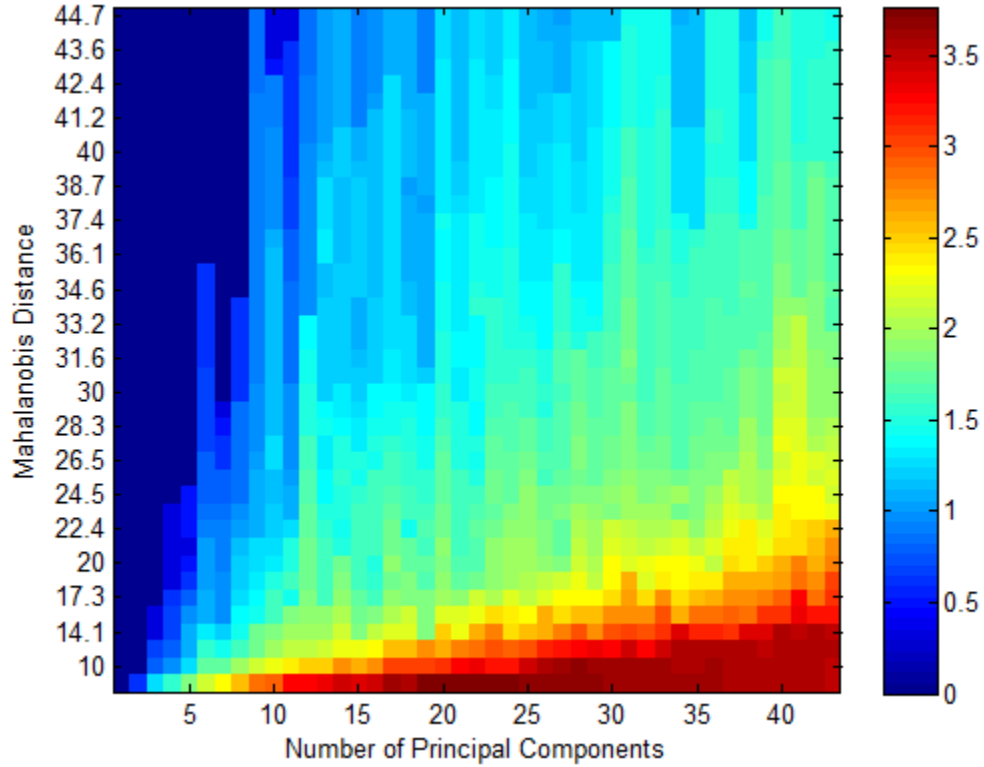


Figure 37: Total Number of Clusters (\log_{10} scale)

Figure 38 shows the CI70 through CL95 metrics on \log_{10} scales across the combinations of Mahalanobis distance and principal components. Again, these plots were generated with delta BIC set to 1. Taken together, the plots indicate that for reasonable values of Mahalanobis distance and principal components, most pixels, presumably from the background, fall into a small number of clusters, which is a desirable behavior. Unexpectedly, the use of the delta BIC with $\lambda = 1$ did not initiate any cluster merges.

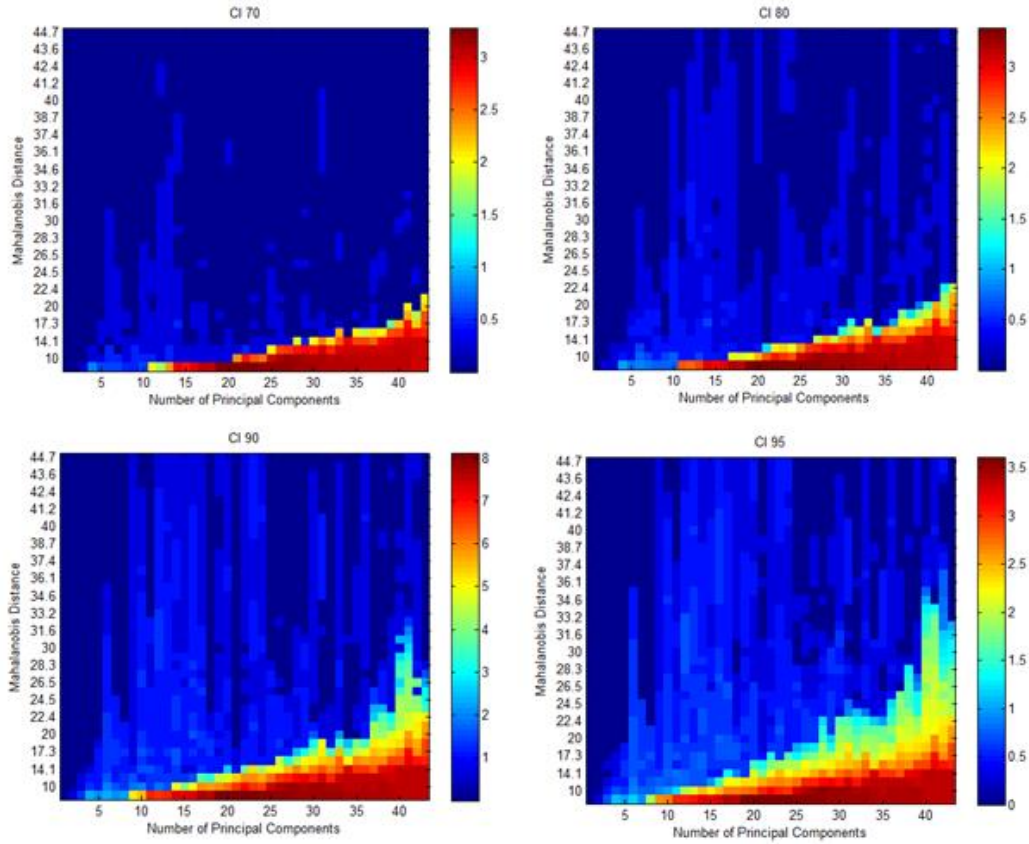


Figure 38: Distribution of Clusters (log₁₀ scale)

The total time required to process the entire image is shown on a log₁₀ (seconds) scale in Figure 39. As expected, when the number of clusters becomes large, as indicated in Figure 37, the time required to execute the algorithm increases accordingly. Despite the creation of clusters throughout the processing of the image, the trend dependence of time on number of clusters is clearly evident at the end of the image.

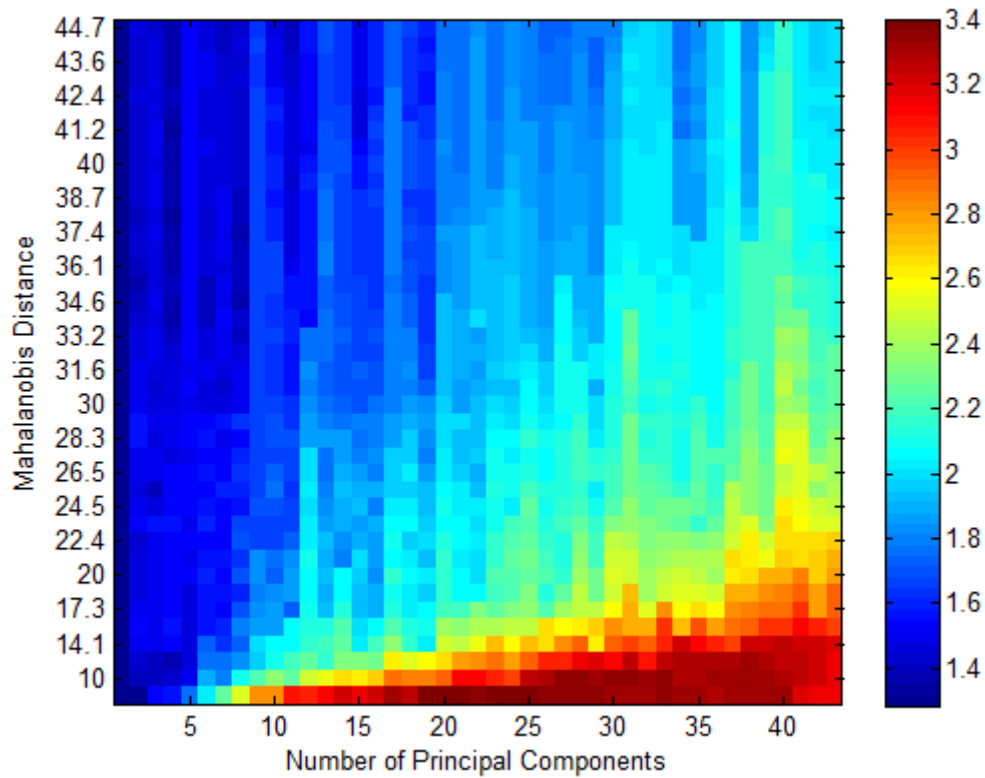


Figure 39: Time required to execute the algorithm in \log_{10} (seconds)

The number of times a pixel was assigned to a cluster based on Mahalanobis distance, when another cluster was closer from a Euclidean distance perspective is shown in Figure 40. It is interesting to note that when the number of clusters becomes exceedingly large, the Euclidean distance rule appears to dominate the assignment of pixels to clusters. Otherwise, substantial differences are seen in the regions where a reasonable number of clusters are formed. This provides further evidence still that Mahalanobis distances are better suited to HSI clustering in principal component space than Euclidean distances.

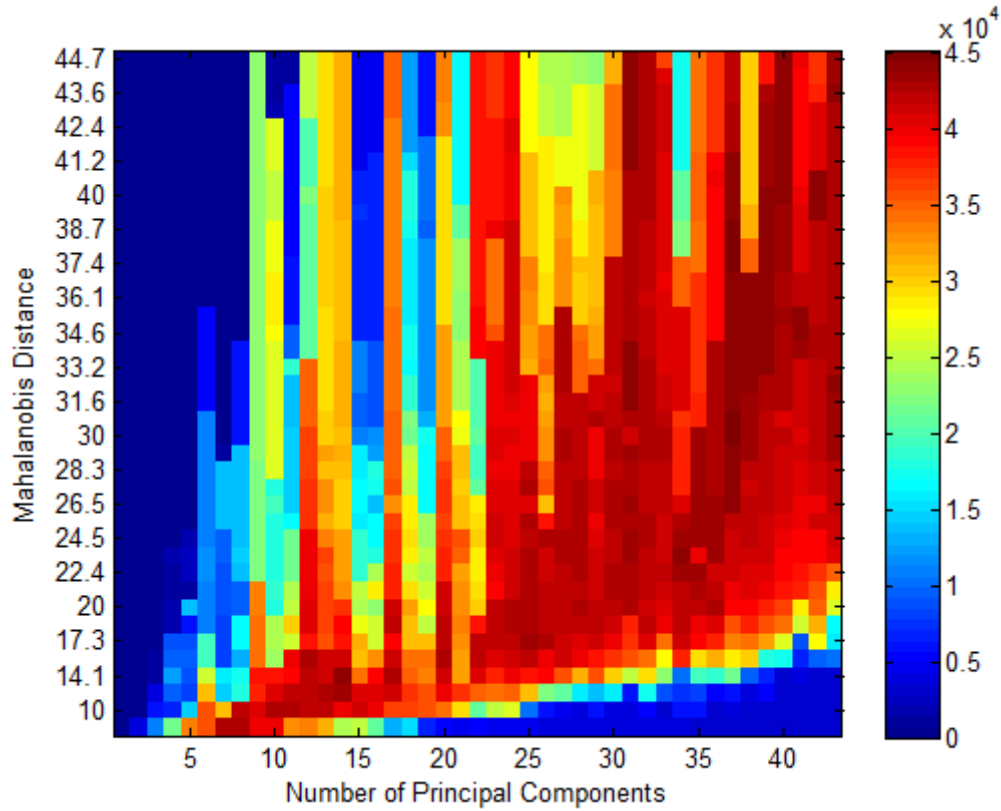


Figure 40: Number of Mismatches between in Mahalanobis and Euclidean Distances

Given the number of detectors evaluated at each of the combinations of Mahalanobis distance and principal components, a review of the maximum, minimum, mean, median and range is appropriate to determine areas of desirable performance. Figure 48 in Appendix B shows these responses for the Lambda=1 case. Generally, the areas where a reasonable number of clusters occur produces behavior in at least one of the 100 detectors that is quite favorable, with an AUC above 0.9. The area of greatest performance appears for the lambda=1 case to be around a Mahalanobis distance of 15 with 15 retained principal components.

Despite the high detector performance in this region, it appears that the recommended settings from the preliminary experiments are somewhat close to the area

of degenerate behavior. Fortunately, changing the value of λ in the delta BIC calculation has a mitigating impact on the degenerate cluster generation behavior which can be seen in Figure 47 in Appendix A. From visual inspection of the responses, a λ value of 15 appears to have the most stable detector performance and clustering behavior on this one image, but a tradeoff exists between detector performance and time required to execute the algorithm. Figure 41 shows the values of AUC and execution time for a subset of the runs and detectors shown in Figure 47. In the case where $\lambda=1$, high detector performance may result in slow clustering. At $\lambda=35$, execution time is consistent, but the number of possible detectors with relatively high performance is lower, and the sensitivity to specific parameters appears to increase.

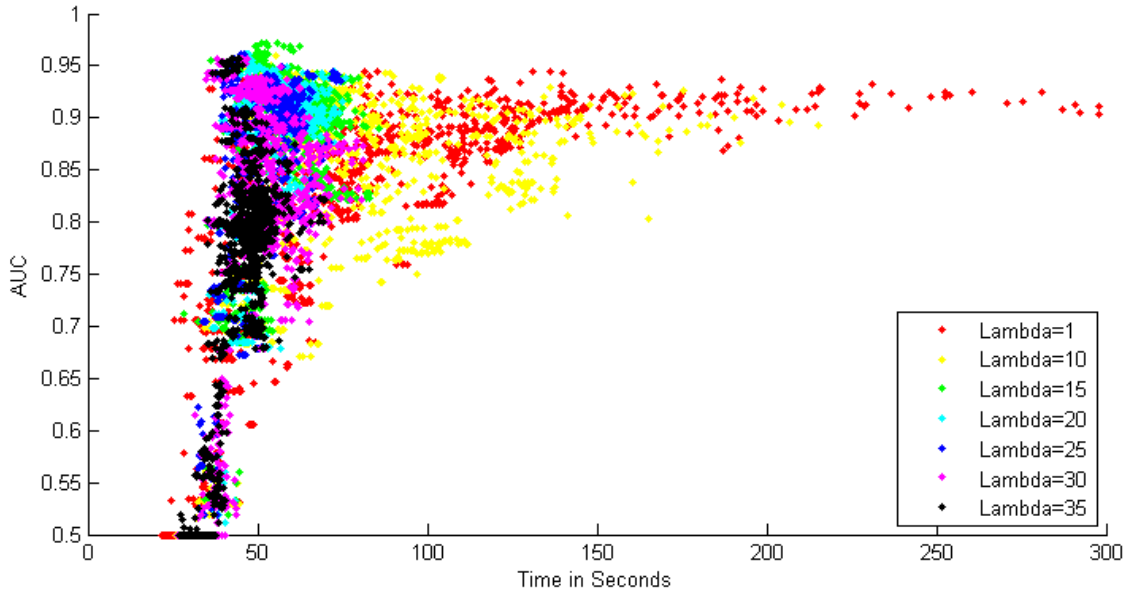


Figure 41: AUC and Execution Time Comparison

By focusing on a the region near the Pareto optimal front in the upper left of the data set, as shown in Figure 42, the case where lambda=15 begins to appear desirable.

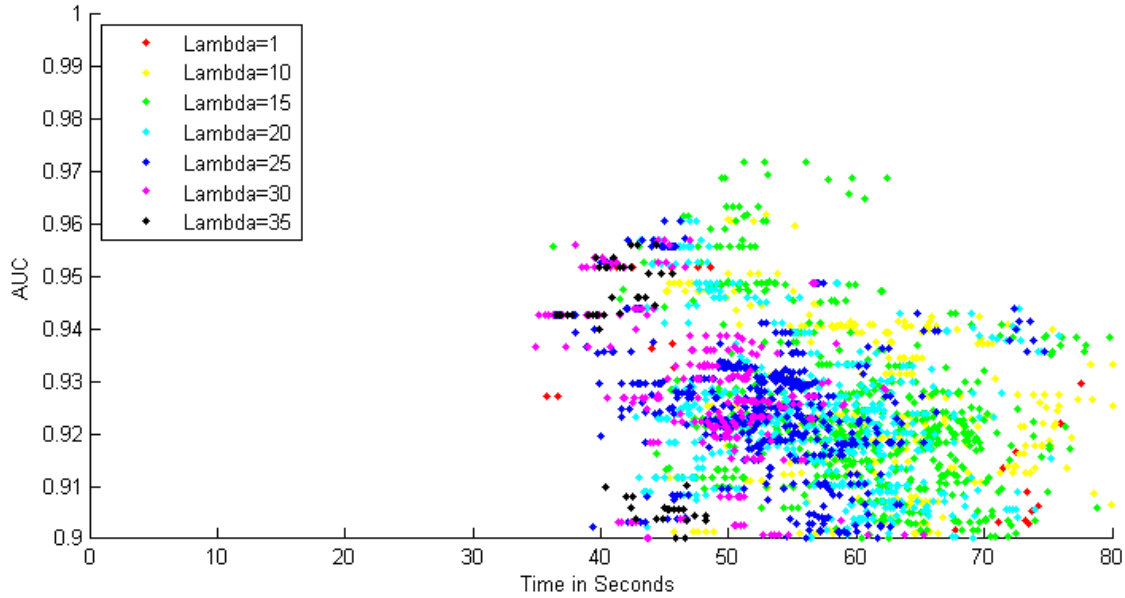


Figure 42: AUC and Execution Time Comparison (Focused Region)

When the stability, or general robustness of the detector settings are subjectively considered, the parameter selections in Table 5 appear to be the suitable, but it is important to remember that only a single image was used in determining these parameters.

Table 5: Single Image Detector Optimal Selected Operating Parameters

Factor Name	Levels
Mahalanbois Distance (squared) Threshold	600
Principal Components Retained	12
Lambda	15
Pixel Memory	8 lines
Anomaly Detection Threshold	16%

Running the online clustering algorithm with these parameters resulted in both poor detector performance and relatively poor clustering behavior. A summary of detector performance is shown in Figure 43. The results of the selected detector (8 lines and 16% threshold) are shown.

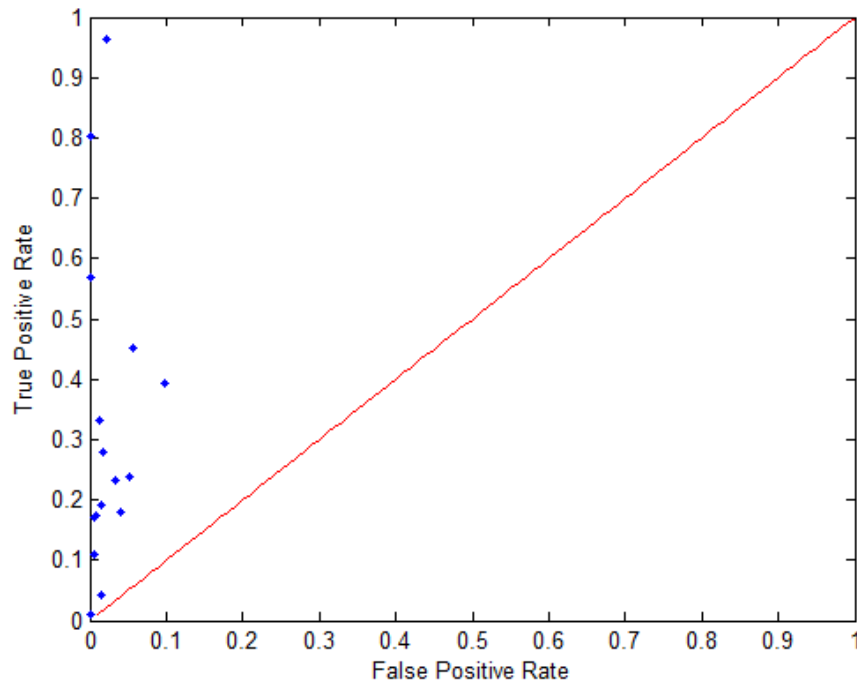


Figure 43: ROC Summary Plot (Table 5 Settings)

The complete results for all images are presented in Appendix C. Figure 49 provides numerical summaries. Figures between Figure 50 and Figure 66, inclusive, provide graphical depictions of the image, the clustering, the anomaly truth mask, a map of the anomalies declared by the algorithm, a direct comparison of the truth mask and the anomalies declared, a ROC curve of the 100 possible detectors used on the image, a histogram of cluster membership, and a plot showing the number of clusters present in the algorithm as the image was processed. For the direct comparison, blue indicates a

true negative, yellow a true positive, orange a false negative and red a false positive. In the anomaly truth mask, yellow indicates an anomaly, blue a background, and red a pixel that has not be determined as either for ground truth (these pixels are not counted toward error rates). Blue and red in the anomaly declaration map denote background and anomaly declarations respectively. The images are based on the detector circled in red on the ROC curve plot.

The 18th image is an AVIRIS image of the Virgin Islands from flight 051219t01 courtesy NASA/JPL-Caltech. The imaging system is different than HYDICE, and has different frequencies and a different number of bands, but the same numbered bands (with their different frequencies) from this imaging system were retained for processing for comparison. Clustering and anomaly detection appear comparable.

Running the algorithm with the parameters suggested by the preliminary experiments in this chapter with a mid-range detector and the standard BIC setting of $\lambda=1$ produced the results shown in Appendix D. Figure 68 summarizes the performance on the individual images and the figures from Figure 69 to Figure 85, inclusive provide the graphical results. Table 6 summarizes the operating parameters.

Table 6: Preliminary Experimentation Suggested Operating Parameters

Factor Name	Levels
Mahalanbois Distance (squared) Threshold	225
Principal Components Retained	15
Lambda	1
Pixel Memory	5 lines
Anomaly Detection Threshold	10%

The detector performance and clustering behavior appears substantially improved over the Table 5 settings, though the algorithm did require slightly more time to execute,

on average, across the various images. The algorithms performance under these parameters demonstrates three major contributions. First, the use of the preliminary experiments to determine likely behavior of this clustering algorithm is shown to be effective. Second the feasibility of clustering a hyperspectral image in the principal component subspace as the image is being collected is shown to be possible. Finally, the concept of anomaly detection as a byproduct of online clustering, is shown to have practical potential.

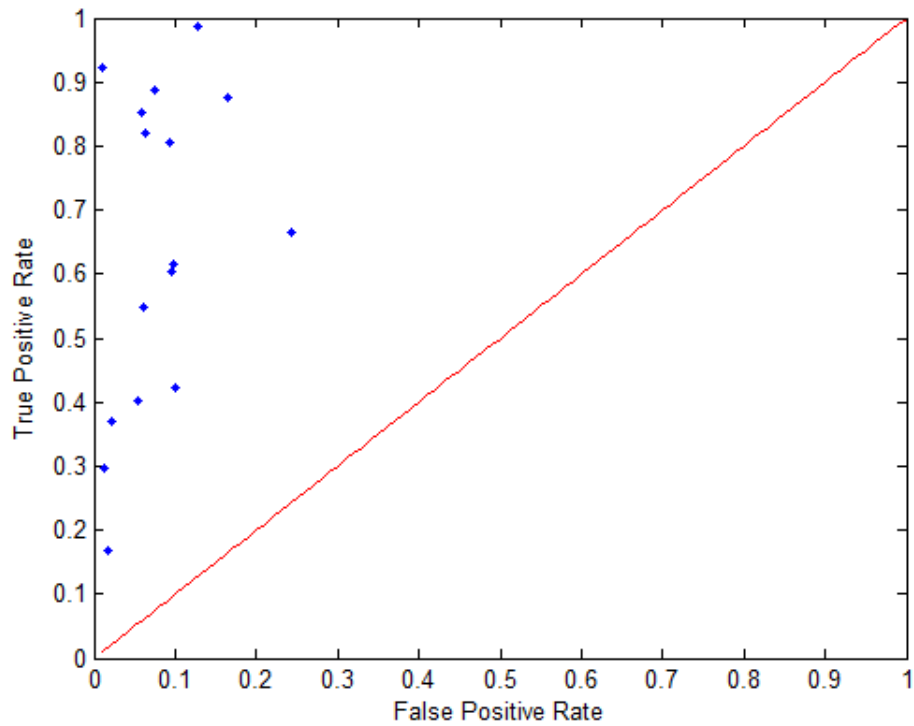


Figure 44: ROC Summary Plot (Table 6 Settings)

Split Cluster Behavior

The split-cluster code was found to be sufficiently computationally intensive as to preclude its detailed exploration, for use in the final code, though a few exploratory

experiments were run with it enabled. Table 7 shows the parameters with split cluster enabled, which produced the clustering in Figure 45 of image 6. This run required just over 73 minutes to complete, in stark contrast to around one minute for processing without the split cluster option enabled. Preliminary behavior with regard to the total number of clusters produced appears to be independent of the setting of the Mahalanobis distance threshold, when both split cluster and merge cluster options are enabled.

Table 7: Settings for Runs with Split Cluster Enabled

Factor Name	Levels
Mahalanbois Distance (squared) Threshold	225
Principal Components Retained	15
Lambda	400
Pixel Memory	5 lines
Anomaly Detection Threshold	10%

The clustering behavior is visually very good. The road on the bottom of the image is well defined across the entire image, and all anomalies are in clusters with other anomalies with similar signatures. The complete set of descriptive plots for this clustering is shown in Figure 87.

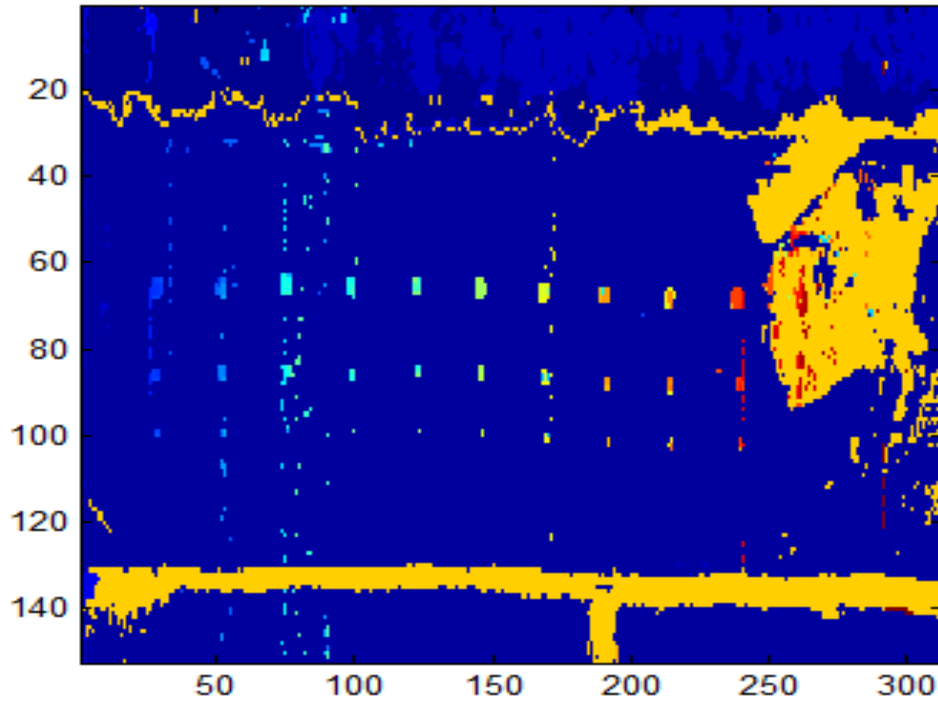


Figure 45: Image Clustering (Table 7 Settings)

Sensitive to Principal Component Selection

While the potential need to calculate the principal components online was mentioned earlier, the problem was essentially assumed away for the purposes of this project. To provide some indication as to the sensitivity of the clustering method to the actual principal components used, an image with a forest background was processed using principal components formed using only sand from a desert image, for a complete mismatch between the principal component space and the image content. The results are shown in Appendix F, Figure 88. A detailed view of the clustering in the image can be seen in Figure 46. The clustering behavior appears to be mostly unaffected by the use of the incorrect principal components, particularly with regard to the clustering of the anomalies in the image.

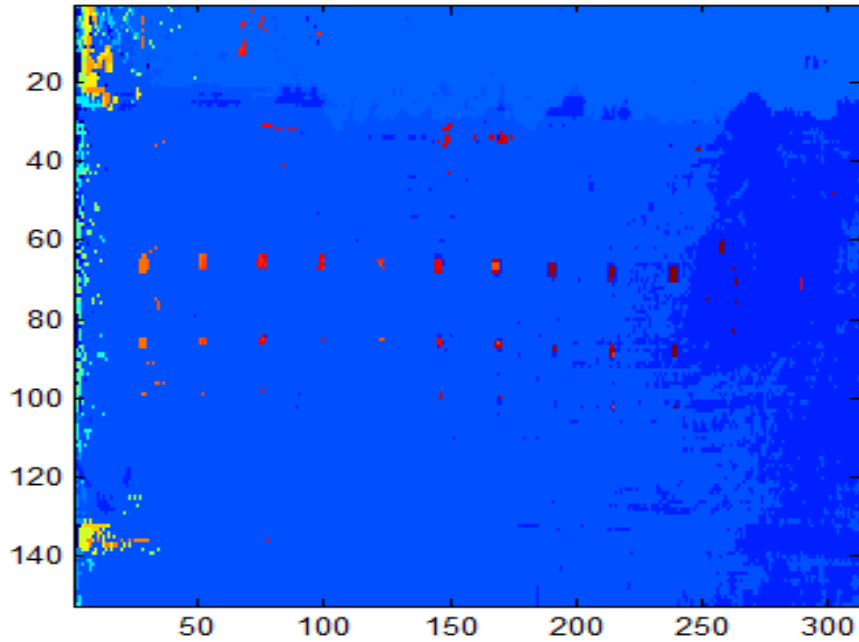


Figure 46: Clustering of Image 6 with Sand Generated Principal Components (Table 7 Settings)

Summary

This chapter has presented the results of the preliminary experiments that showed both the inadequacy of the Euclidean distance for this clustering application, and the stability of the behavior of the Mahalanobis distance. The experiments also produced a set of parameters likely to produce desirable clustering behaviors. These parameters produced visually reasonable clustering behaviors when used across 17 typical hyperspectral images. An optimization using a single image produced remarkable detection accuracy for anomalies in that image, but failed to produce a robust set of parameters for use in other images for both clustering and anomaly detection. While the performance of the rudimentary detectors demonstrated was not eye watering, it was sufficient to demonstrate the feasibility of anomaly detection as a byproduct of the online clustering algorithm.

V. Conclusions and Recommendations

Chapter Overview

This chapter highlights the key results and contributions of this research project and discusses the significance of those contributions. It then proceeds to recommend a number of areas for future research consideration.

Conclusions of Research

The first major conclusion of this project is the superiority of Mahalanobis distance to Euclidean distance when clustering hyperspectral imagery pixels in the principal component subspace. More generally, by defining a repeatable set of experiments for exploring the behavior of common signatures in the subspace, and demonstrating the efficacy of their results, this project has shown that is possible to evaluate certain characteristics of a domain and determine an estimate of the types of distances and parameter settings that could be used effectively for this type of clustering algorithm before actually using the algorithm on data from the domain itself. This would allow the extension of the online algorithm to other domains, such as computer networking, with a solid estimate of operating parameters in those domains after the generation of a few basic plots using some example data that might be available. A byproduct of this aspect to the research was the identification of specific anomaly behavior in the principal component subspace that could be used to develop both image analysis tools and potentially a class of ensemble detectors.

The second major conclusion of this project is the feasibility of a reasonably fast online clustering algorithm for use with hyperspectral imagery. While somewhat

sensitive to the parameter settings, it is clearly possible to perform clustering as the image is collected, and to use this clustering to support anomaly detection

The final conclusion of this project is the potential to perform anomaly detection as a byproduct of clustering in an online clustering algorithm. The implementation of a set of rudimentary detectors, based on the behavior of the online clustering algorithm, lays the foundation for further cluster based outlier work in hyperspectral imagery.

Significance of Research

As discussed in Chapter 2, operational considerations drive the need for a method to detect anomalies in hyperspectral imagery as the image is being collected. Common detection methods are improved by the use of clustering, but an online detection method requires an online clustering algorithm to reap the benefits provided by clustering. This project has shown both the feasibility of online clustering in hyperspectral imagery, and the feasibility of a rudimentary class of detector based on clustering. Further, key observations about the behavior of common hyperspectral signatures in the principal component subspace can support future detector and image processing work, as well as potential detector development for other domains.

Recommendations for Future Research

Several areas are recommended for future research based on the results seen in this project.

One of the main areas of future research would be the refinement of detector and clustering parameters, potentially through the use of a robust parameter designed experiment. Also, the numerous measures of performance for the algorithm suggest that

some work should be done toward determining the relative importance of speed, the various components of accuracy, and other potentially important characteristics such as desirable number of clusters produced by the algorithm. Particularly with regard to the desired number of clusters, this project has performed a demonstration which sought to also place anomalies into separate clusters from the background. If this task is not a requirement, or useful for a given application, the possibility of clustering in a much lower dimensional space, perhaps as few as two principal components, should be explored. Additionally, a separate data set with known clustering would allow for an evaluation of the clustering accuracy.

Another area of suggested work, is to increase the efficiency of the algorithm to reduce the execution time. Several possibilities are immediately evident.

- As implemented, the algorithm does not store the natural log of the determinant of the covariance matrix, resulting in the recalculation of this value multiple times as each pixel is read. Calculating this value a single time then storing it would be preferable, and would save an $O(d^3)$ determinant operation.
- As implemented, a linear search across all clusters occurs after each pixel is read. If the clusters could be indexed into a tree structure, the $O(c)$ search step could be reduced to $O(\log(c))$ step. Additionally, given the spatial correlation present in images, and the potential for clear separation between clusters, the search could be designed to start with the cluster to which the last point was added, and could stop at the first cluster where the Mahalanobis distance falls below a certain threshold. If similar or improved clustering results could be achieved with this method, the speed-up would be substantial.
- When implemented, the use of k-means to check whether or not to split a cluster at each step becomes computationally prohibitive. Using k-means at various intervals, or at key points to check clusters (such as when the cluster first doubles in size after forming a valid covariance matrix), could reduce the computational cost but still retain the benefits of including a split cluster step.

- Lastly, the deletion of pixels as they “age out” during the algorithm could both change the behavior of the algorithm and speed up the use of k-means when assessing a cluster for splitting. This feature should be fully implemented in the algorithm in the future.

The behavior of the Euclidean rules used prior to the formation of a valid covariance matrix could be explored further, and an alternative method might be possible. Euclidean distances have been demonstrated to be inappropriate for clustering in the space, so an alternative initial distance measure might improve performance overall.

Another related area would involve the exploitation of having various outliers in the same cluster. Spatial methods of anomaly detection, which seek to use both context and spatial data to find anomalous features, could potentially reap a benefit from having anomalous pixels with similar signatures, already grouped together spectrally. Were a stereo or multi-static hyperspectral dataset available, the clustering could even be potentially used as a consistency measure that takes advantage of spatial correlation in images for voxel coloring as described by (Waddell, 2003).

For this project, a global principal component analysis has been used, under the assumption that either an online PCA method could be implemented to address issues with changes occurring in the image that might change the principal component structure. This area should be addressed in future work. A closely related area is the use of local rather than global dimensionality reduction, which might allow each cluster to have a separate set of principal components. Alternatively, a single set of principal component transforms applicable to a wide range of images might be possible, and should be further explored.

For clustering, it does not appear that the DBSCAN algorithm has been implemented for HSI processing. A feasibility demonstration of this algorithm to this domain could prove useful.

The algorithm should be assessed for applicability to other domains, such as anomalous process detection in computer security applications where speed is important and online clustering may be useful. The demonstration of the utility of the Mahalanobis distance measure for use in computer security by (Shilland, 2009), gives an indication that the algorithm might be applicable to this domain.

The image processing tool suggested in Chapter 4 should be implemented and used to explore the behavior in the higher dimension principal components. This may yield a new method of anomaly detection in the principal component subspace.

Summary

This chapter has reviewed key results and contributions of this research project and made numerous recommendations for further research in the area of online clustering and anomaly detection in hypersepctral imagery.

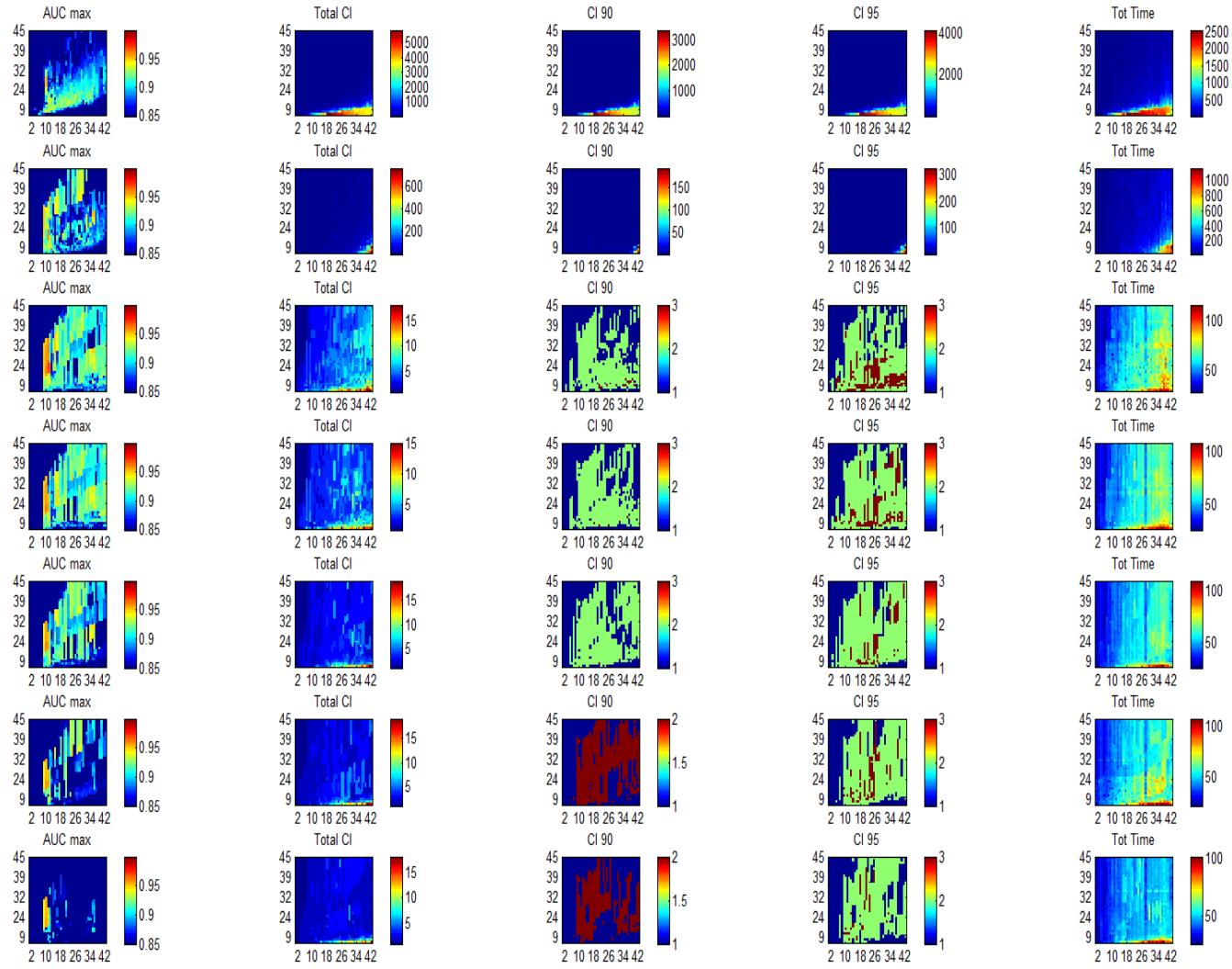


Figure 47: Summary Results for $\Lambda=1, 10, 15, 20, 25, 30, \& 35$

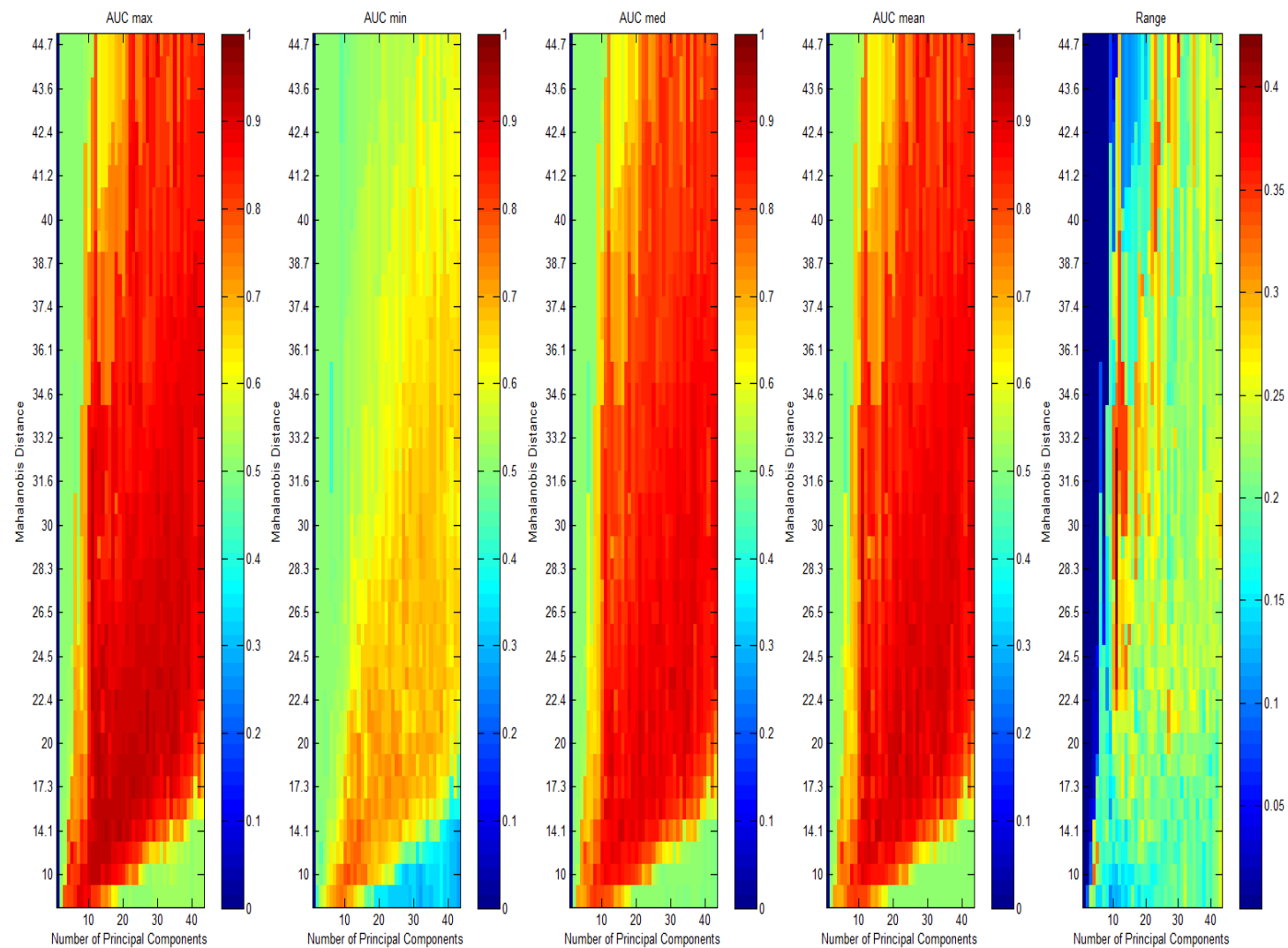


Figure 48: Detector Performance for Lambda=1

Appendix C

Image#	Ctot	C70	C80	C90	C95	C96	C97	C98	C99	EM Miss	Merges	Splits	Assign Time	Split Time	Combine Time	Total Time	TPR	FPR	TNR	FNR	Height	Width	Total Pixels	Actual # of Anomalies
1	1	1	1	1	1	1	1	1	1	0	6	0	3.81	1.07	0.25	5.13	N/A	0.00	1.00	N/A	108	203	21924	0
2	2	1	2	2	2	2	2	2	2	5172	5	0	15.97	7.07	0.68	23.72	0.17	0.01	0.99	0.83	199	291	57909	1109
3	2	1	1	1	2	2	2	2	2	959	7	0	7.70	2.06	0.35	10.12	0.18	0.04	0.96	0.82	160	191	30560	2953
4	1	1	1	1	1	1	1	1	1	0	3	0	4.40	1.35	0.28	6.02	N/A	0.00	1.00	N/A	124	198	24552	0
5	7	2	3	3	4	4	4	4	4	11300	7	0	11.01	1.15	0.26	12.42	0.45	0.06	0.94	0.55	104	215	22360	4407
6	3	1	1	2	2	2	2	2	2	7510	8	0	17.08	4.76	0.56	22.39	0.96	0.02	0.98	0.04	152	312	47424	2749
7	2	1	1	1	2	2	2	2	2	1426	2	0	6.28	1.33	0.28	7.89	0.04	0.01	0.99	0.96	156	156	24336	748
8	2	1	1	1	1	1	1	1	2	850	6	0	2.73	0.33	0.13	3.19	0.80	0.00	1.00	0.20	104	106	11024	381
9	5	1	1	1	1	1	1	1	1	3	1	0	1.05	0.07	0.05	1.18	0.57	0.00	1.00	0.43	61	73	4453	175
10	3	1	1	2	2	2	2	2	3	5274	8	0	8.56	2.07	0.35	10.98	0.33	0.01	0.99	0.67	136	226	30736	773
11	5	1	2	2	2	2	2	2	2	6229	3	0	13.20	2.76	0.42	16.38	0.28	0.02	0.98	0.72	78	460	35880	3930
12	4	1	1	1	1	1	1	1	2	911	11	0	3.92	0.64	0.20	4.76	0.19	0.01	0.99	0.81	80	205	16400	787
13	6	1	2	3	3	3	3	4	5	6562	15	0	20.53	5.86	0.65	27.04	0.39	0.10	0.90	0.61	150	355	53250	2667
14	4	2	2	2	2	2	2	2	2	1938	8	0	3.24	0.24	0.11	3.59	0.17	0.01	0.99	0.83	68	139	9452	825
15	4	1	1	1	2	2	2	3	3	12693	12	0	28.14	10.82	0.86	39.81	0.23	0.03	0.97	0.77	155	470	72850	4405
16	3	2	2	2	2	2	2	2	2	3255	7	0	5.42	0.65	0.19	6.26	0.11	0.01	0.99	0.89	77	215	16555	586
17	2	1	2	2	2	2	2	2	2	1289	7	0	3.66	0.49	0.16	4.31	0.01	0.00	1.00	0.99	88	161	14168	968
18	4	3	3	3	4	4	4	4	4	13643	2	0	10.54	1.24	0.27	12.05	0.24	0.05	0.95	0.76	163	144	23472	80

Figure 49: Summary of Clustering Performance (Table 5 Settings)

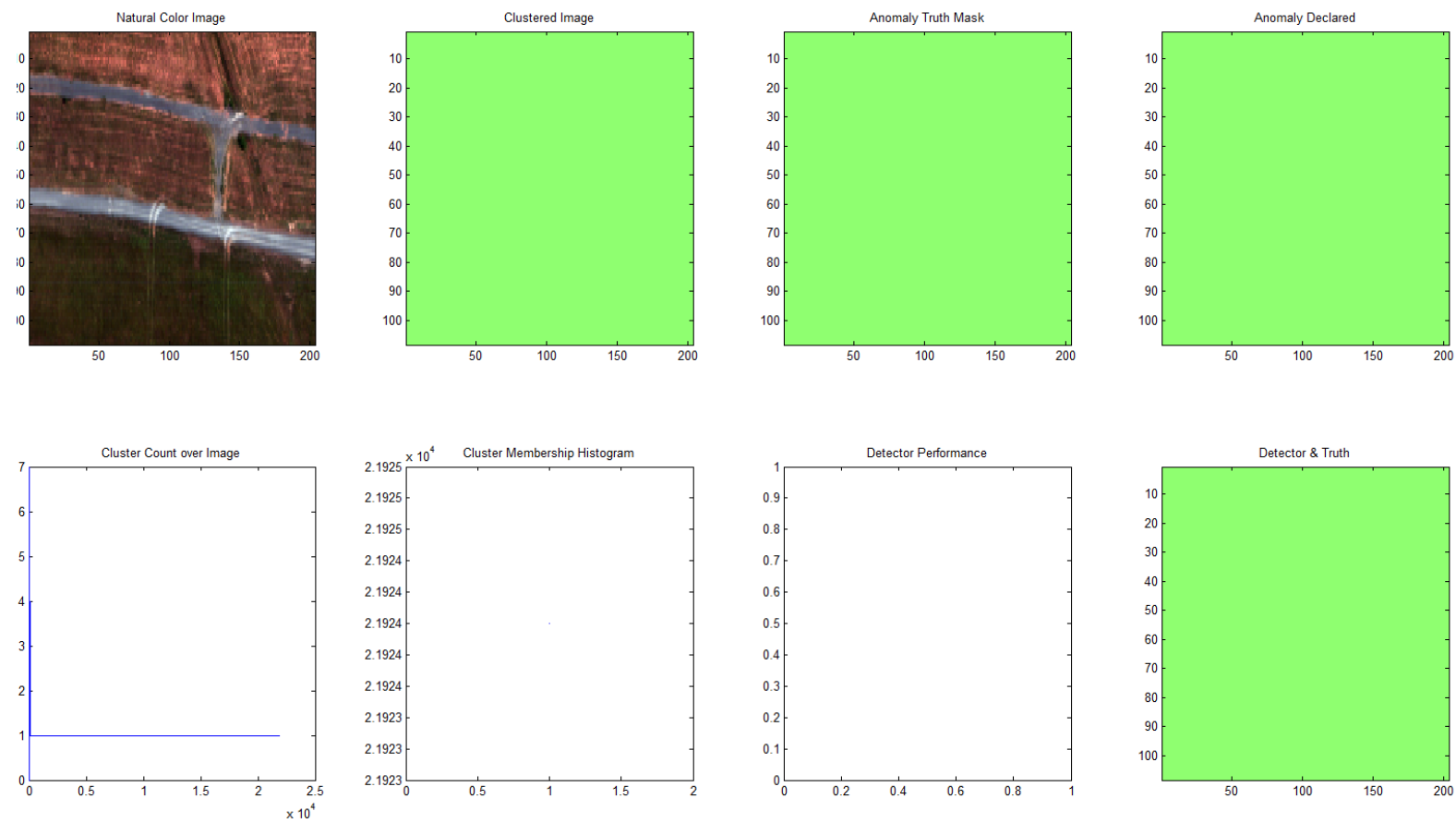


Figure 50: Image 1 Results (Table 5 Settings)

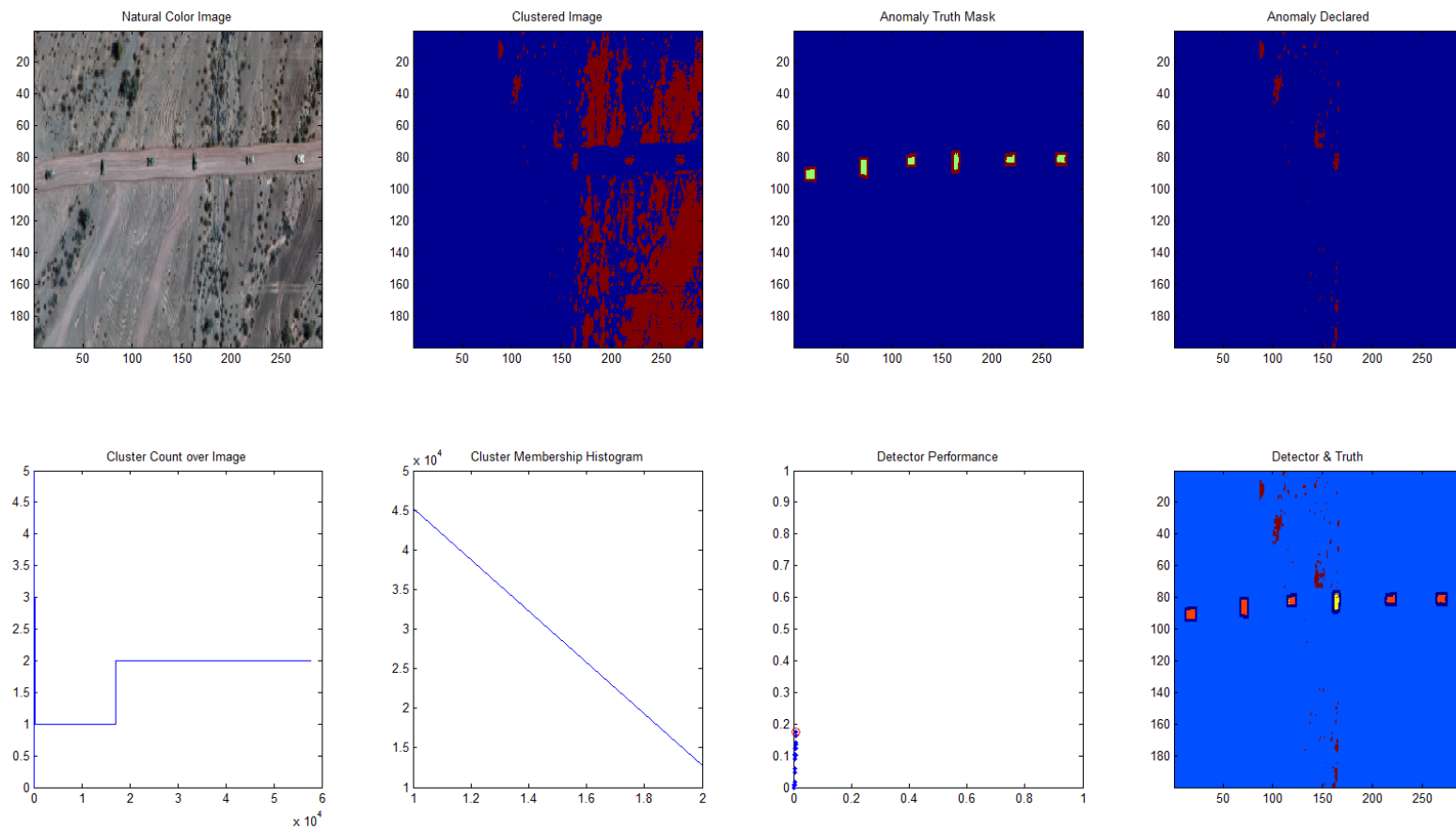


Figure 51: Image 2 Results (Table 5 Settings)

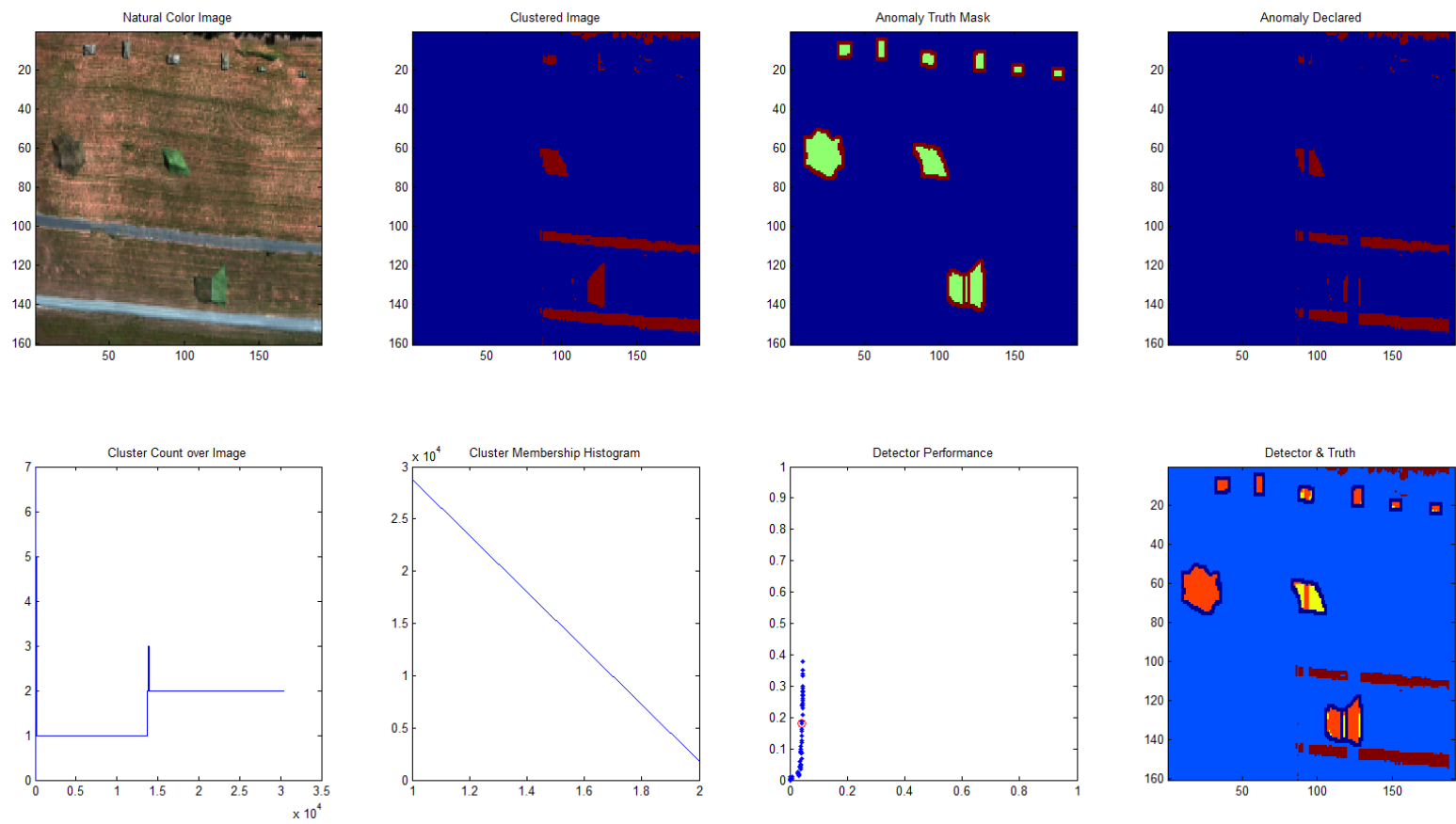


Figure 52: Image 3 Results (Table 5 Settings)

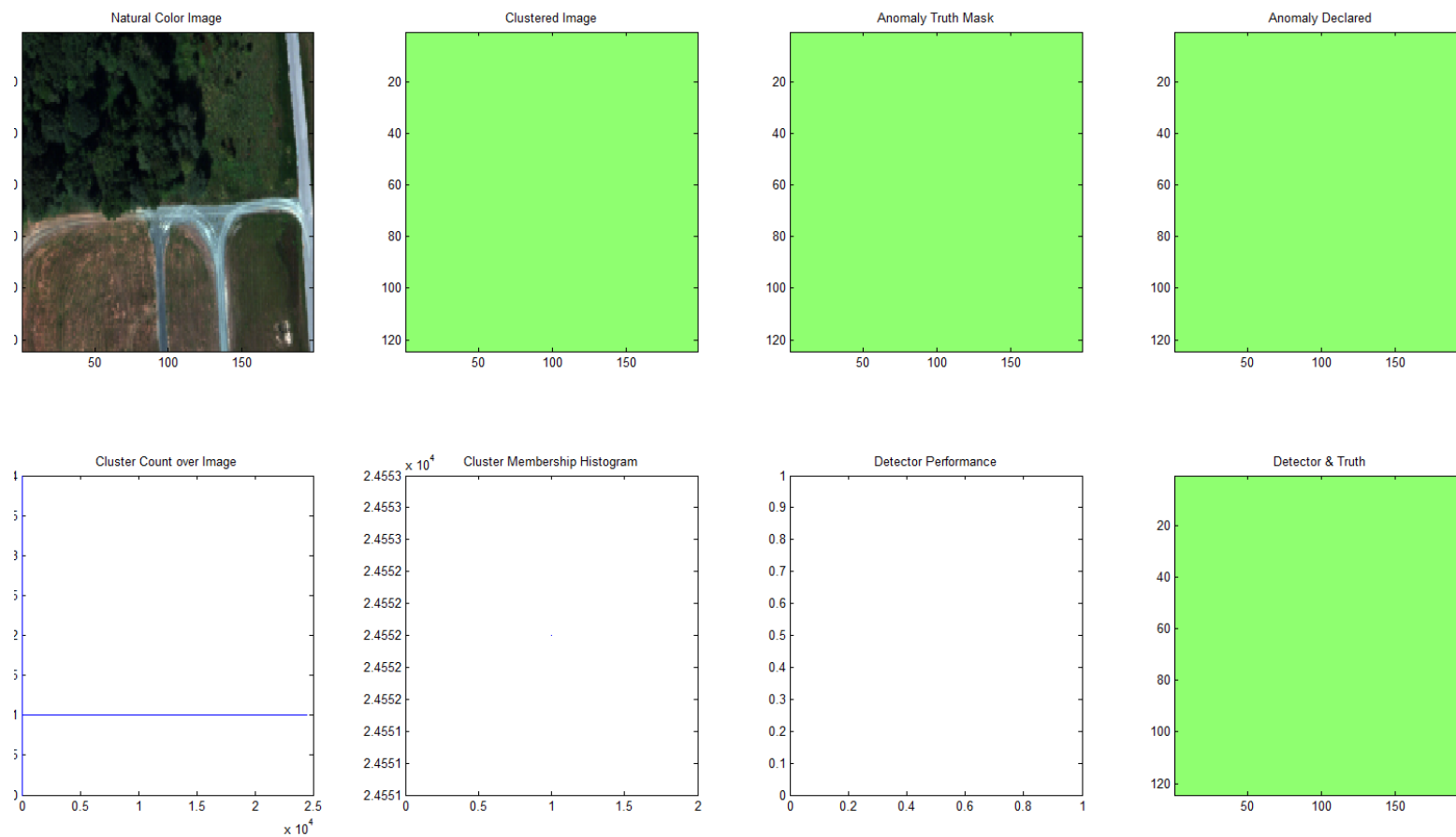


Figure 53: Image 4 Results (Table 5 Settings)

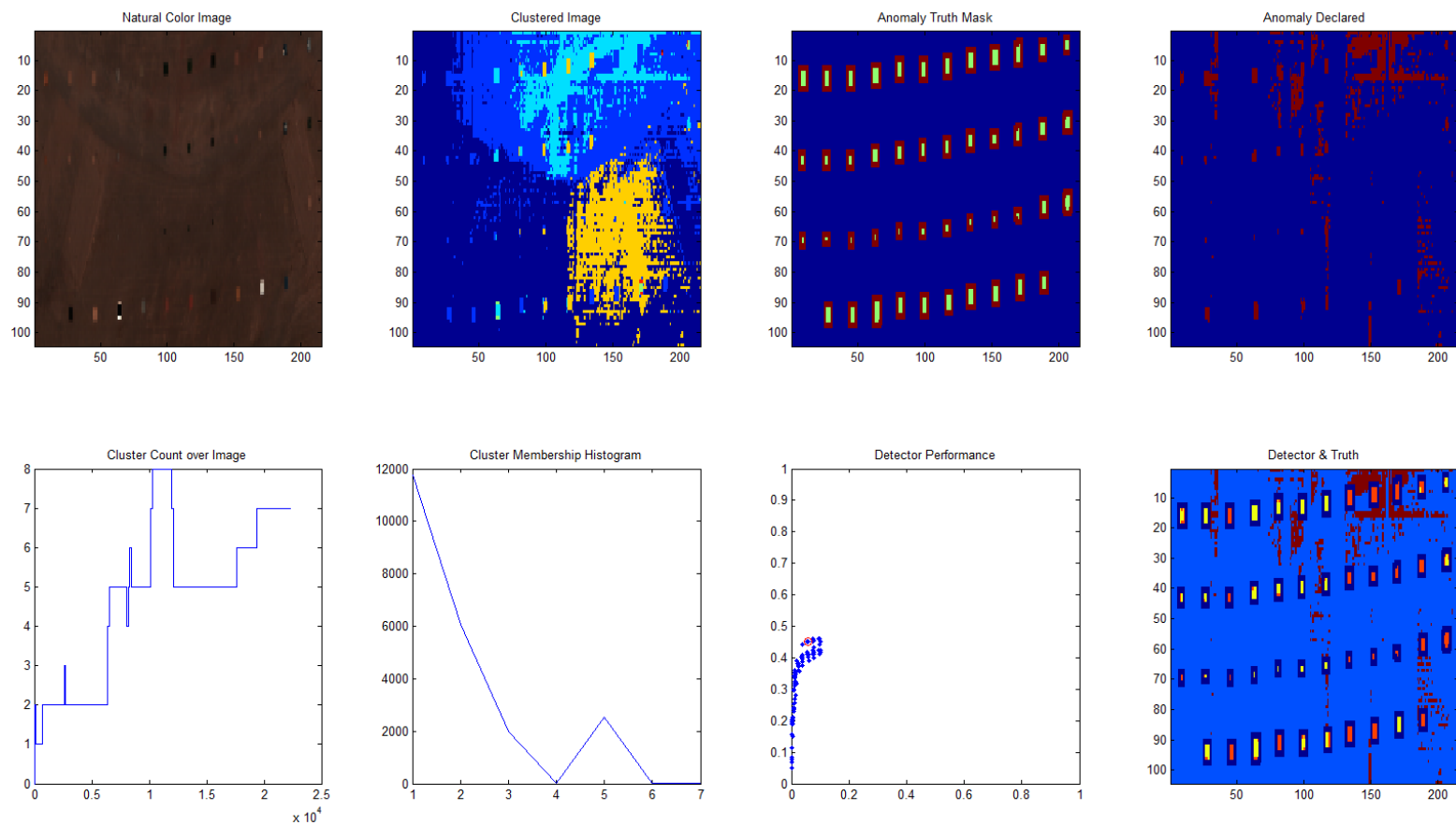


Figure 54: Image 5 Results (Table 5 Settings)

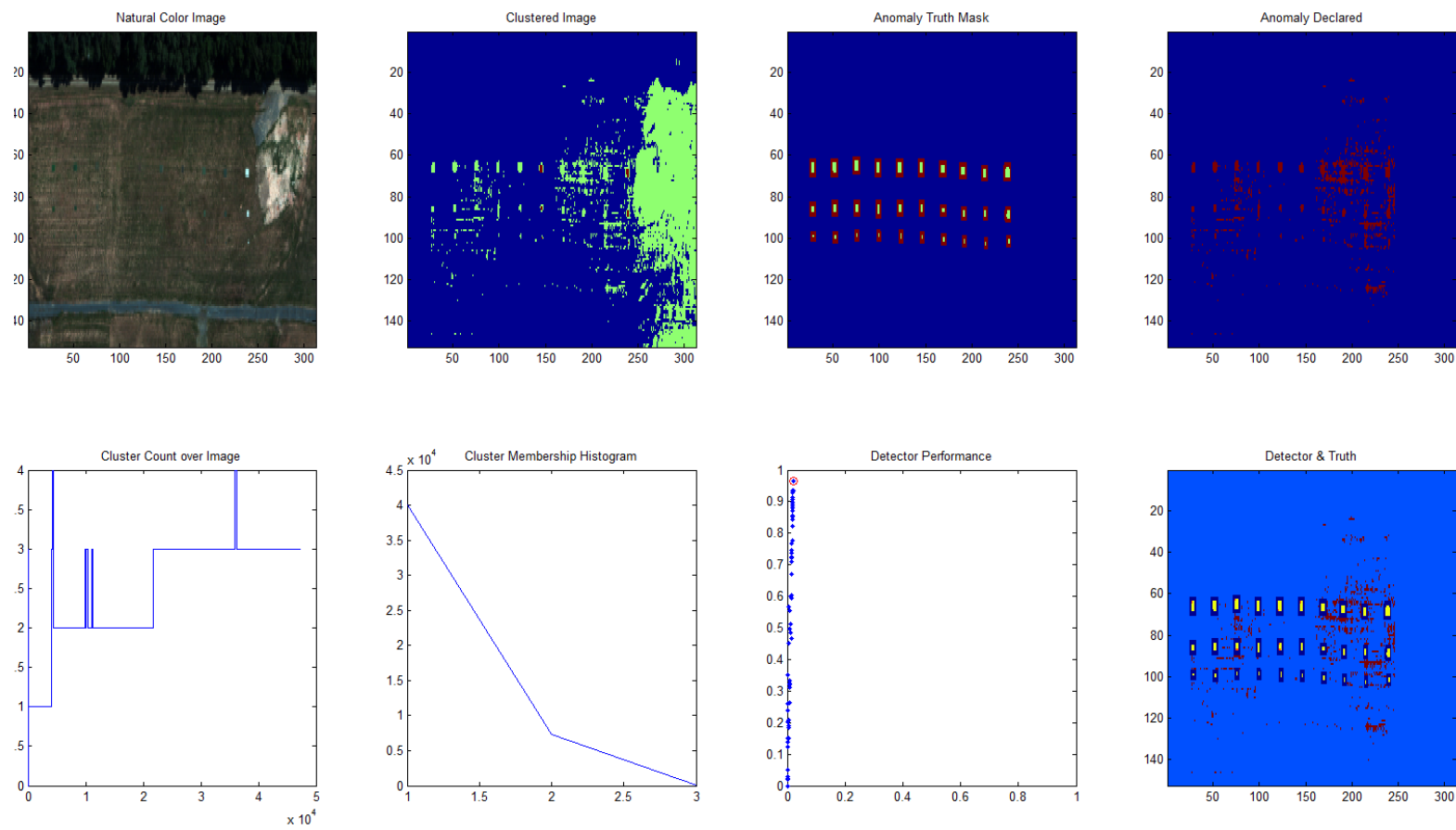


Figure 55: Image 6 Results (Table 5 Settings)

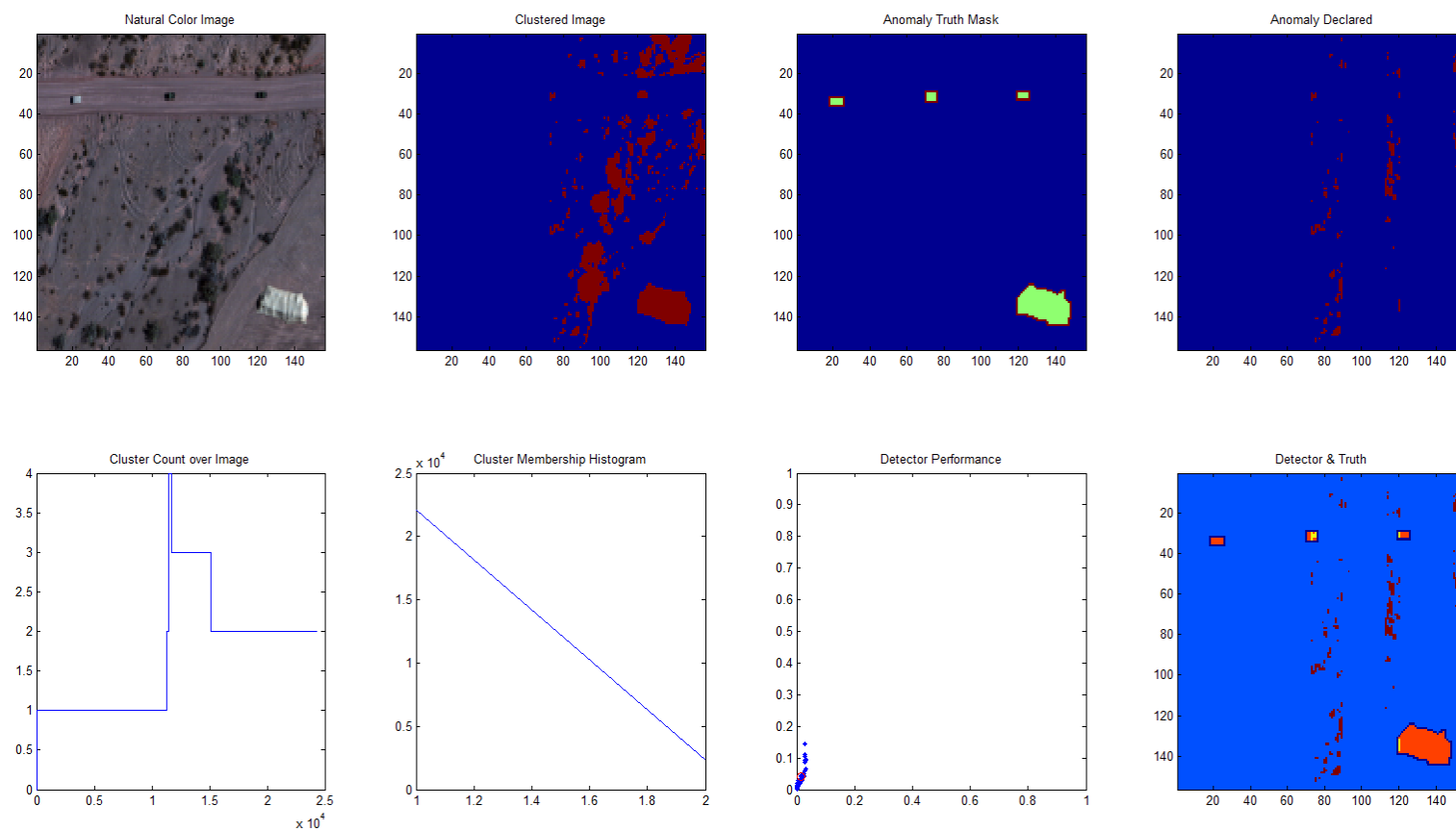


Figure 56: Image 7 Results (Table 5 Settings)

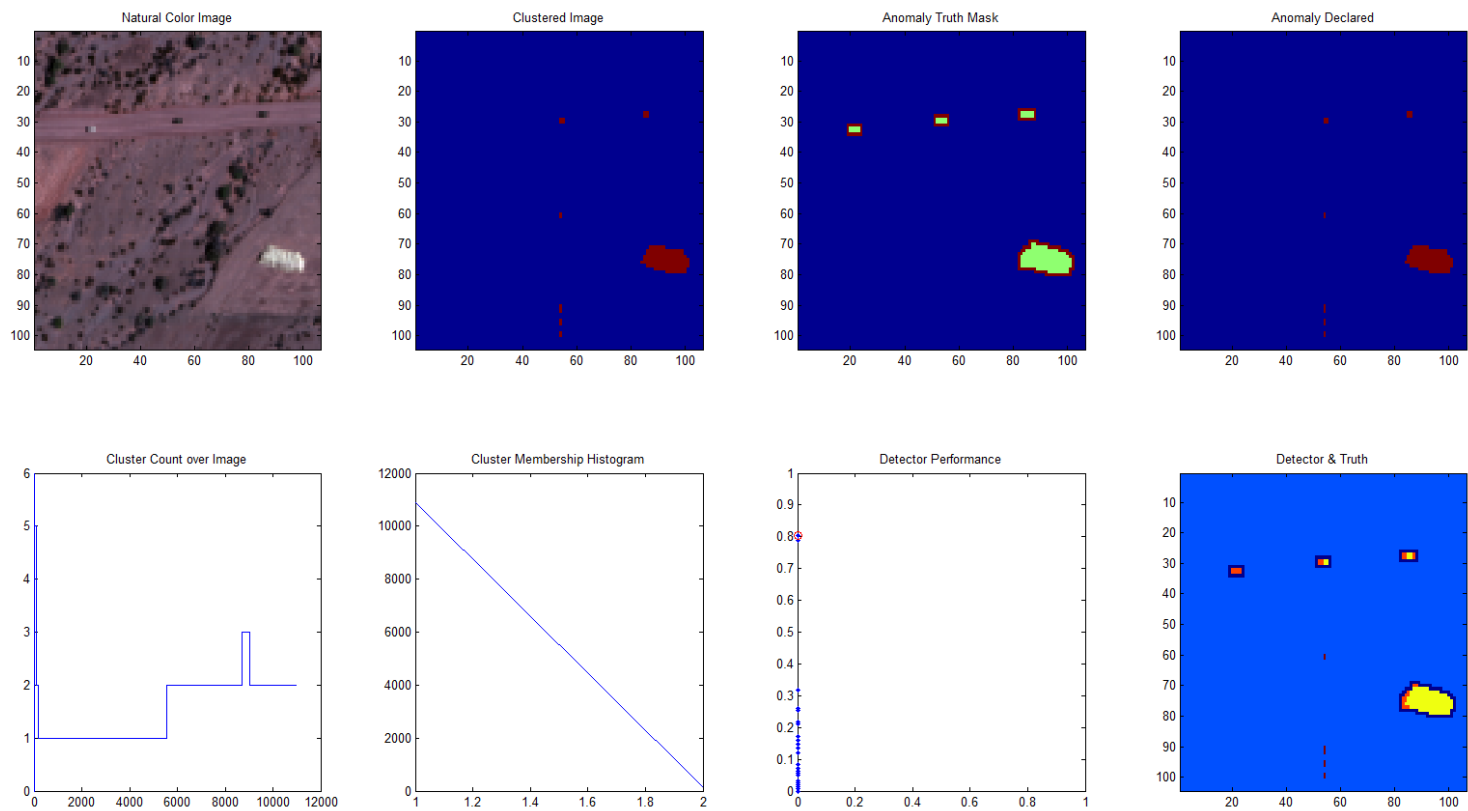


Figure 57: Image 8 Results (Table 5 Settings)

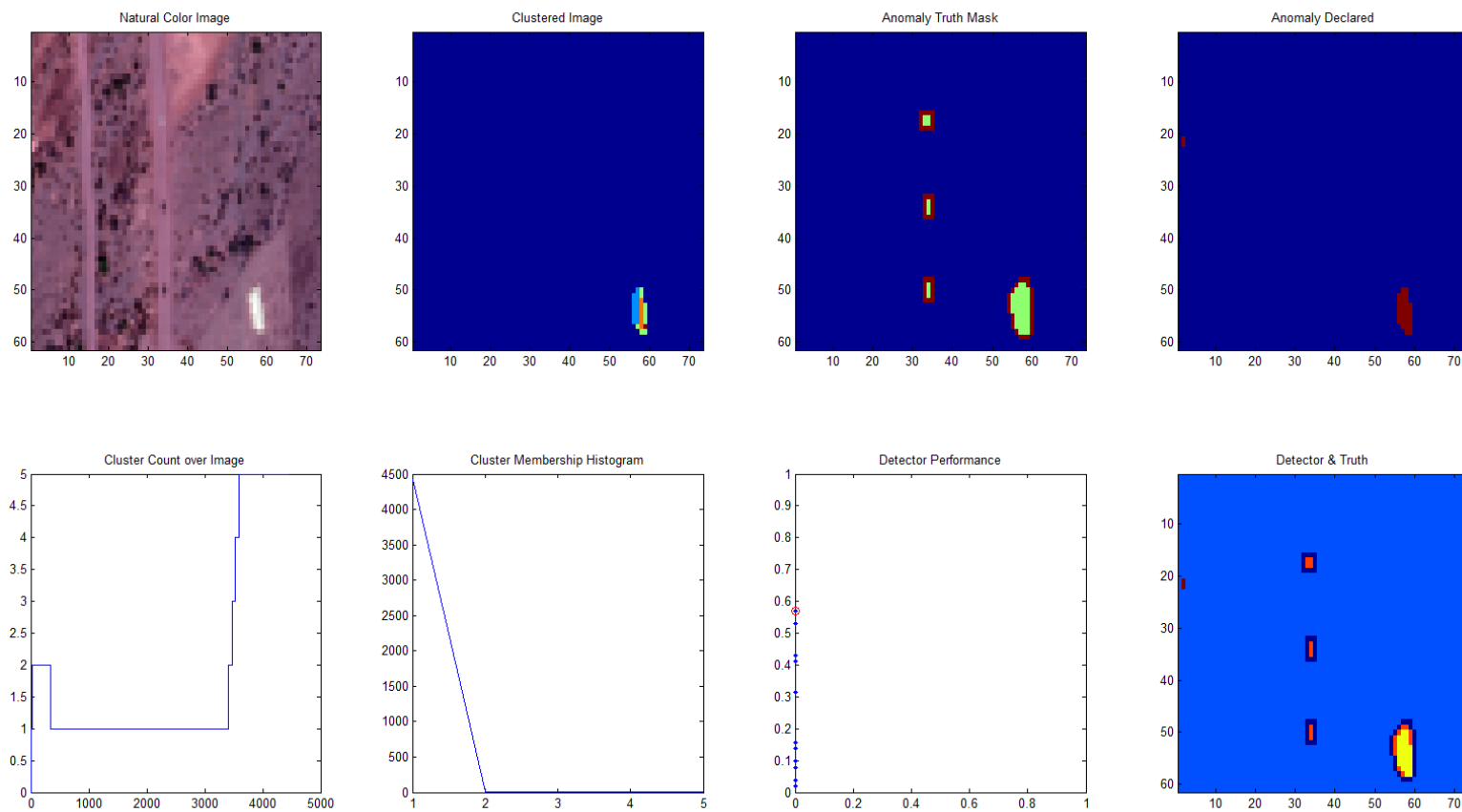


Figure 58: Image 9 Results (Table 5 Settings)

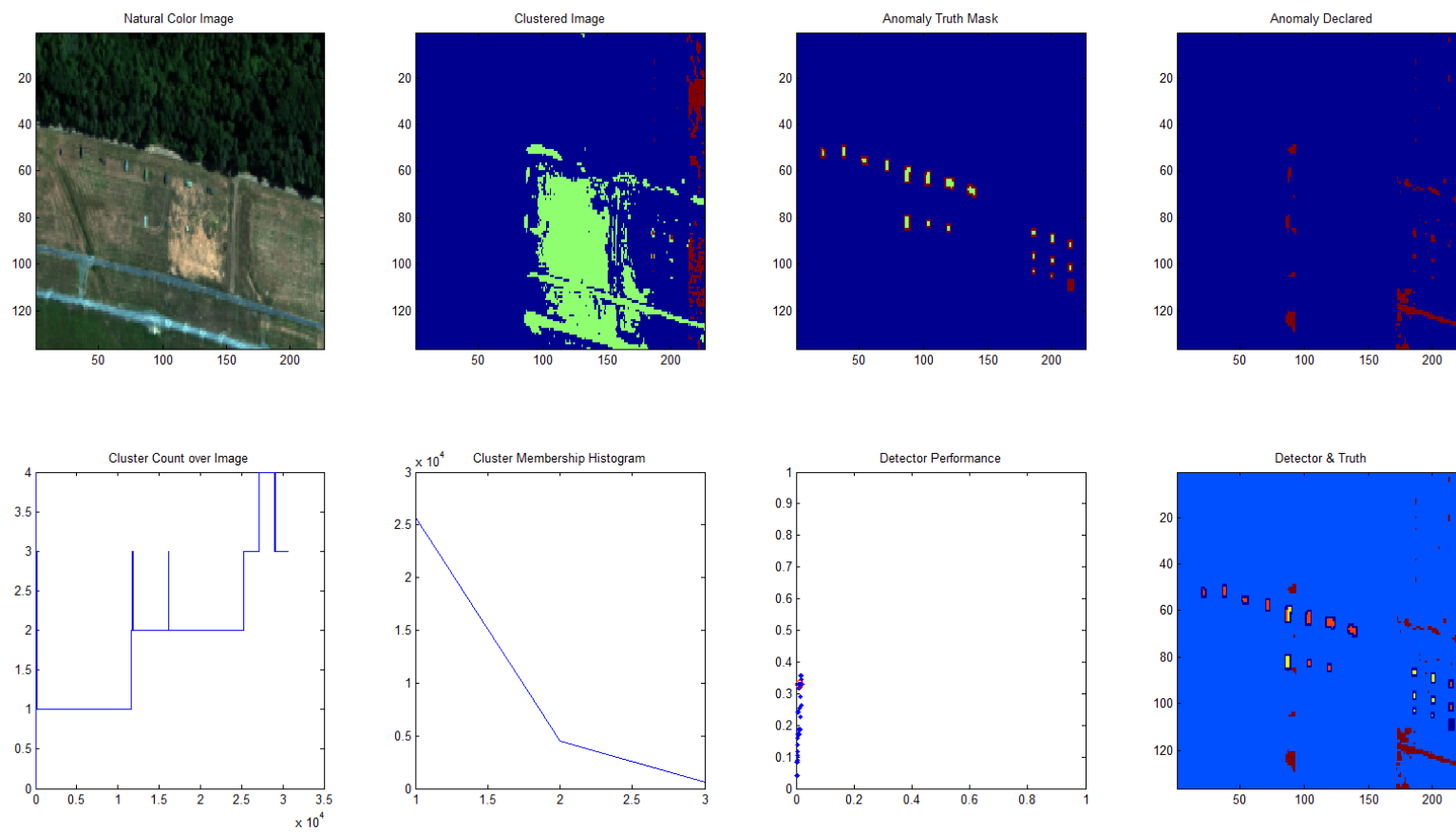


Figure 59: Image 10 Results (Table 5 Settings)

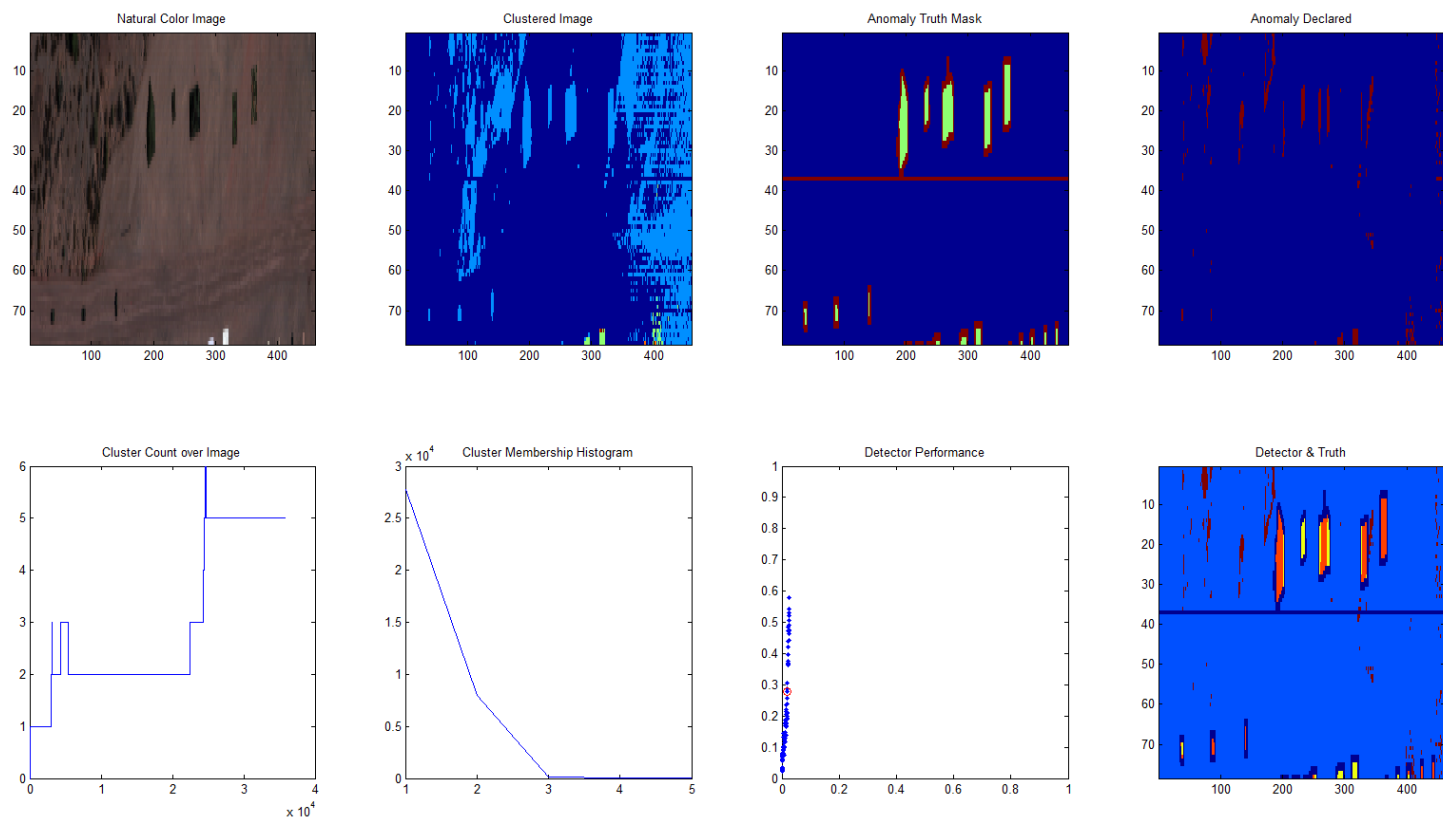


Figure 60: Image 11 Results (Table 5 Settings)

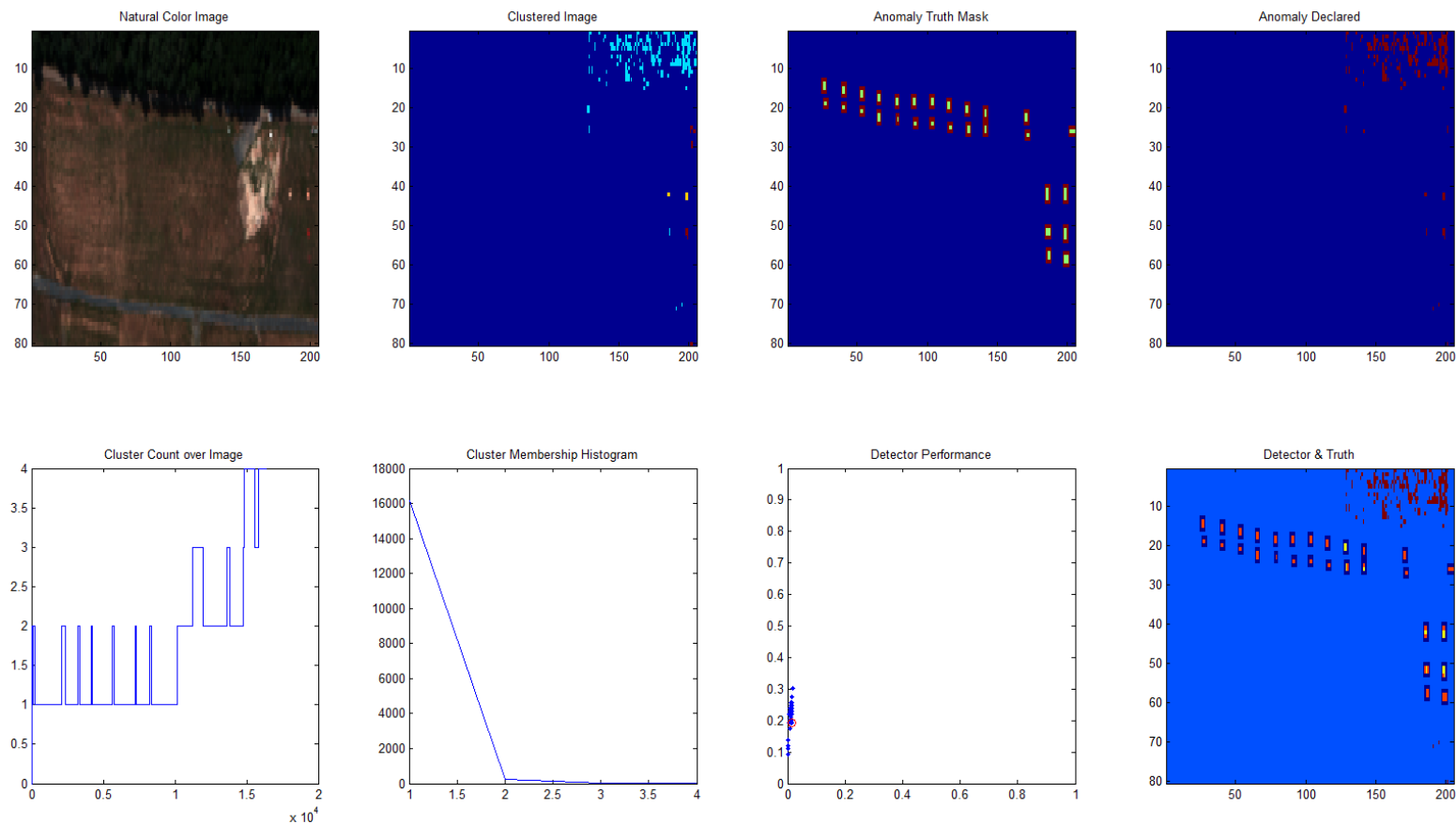


Figure 61: Image 12 Results (Table 5 Settings)

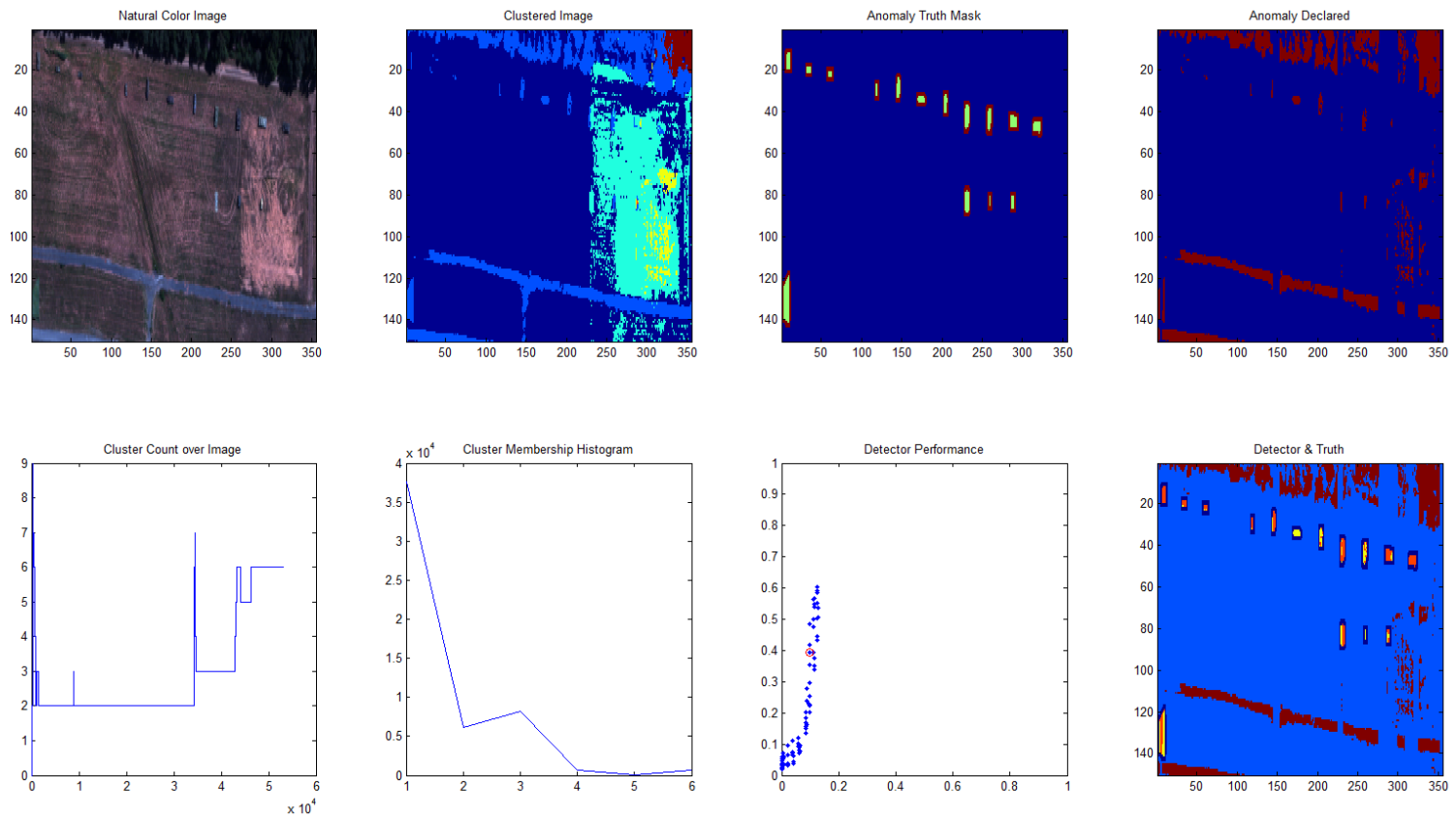


Figure 62: Image 13 Results (Table 5 Settings)

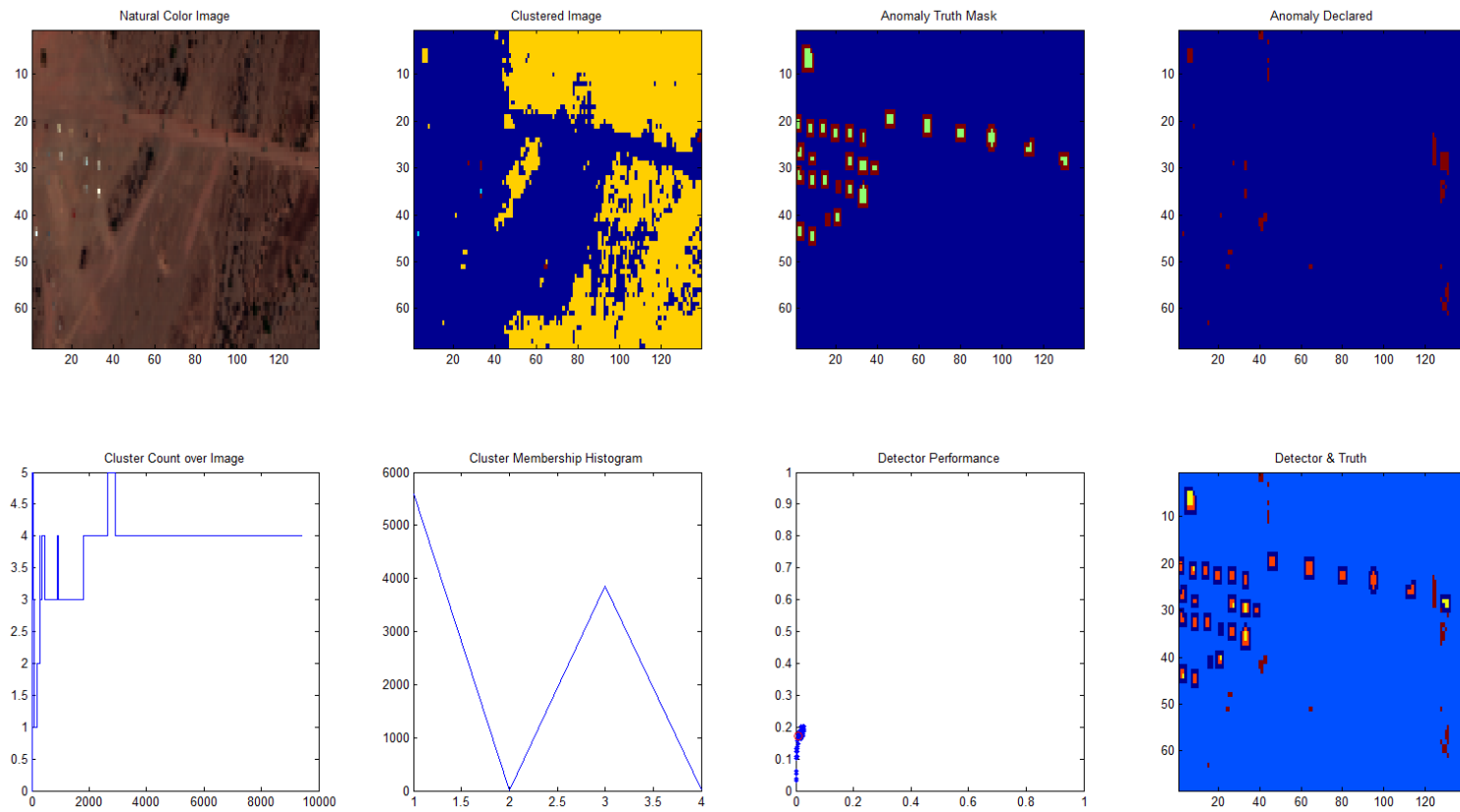


Figure 63: Image 14 Results (Table 5 Settings)

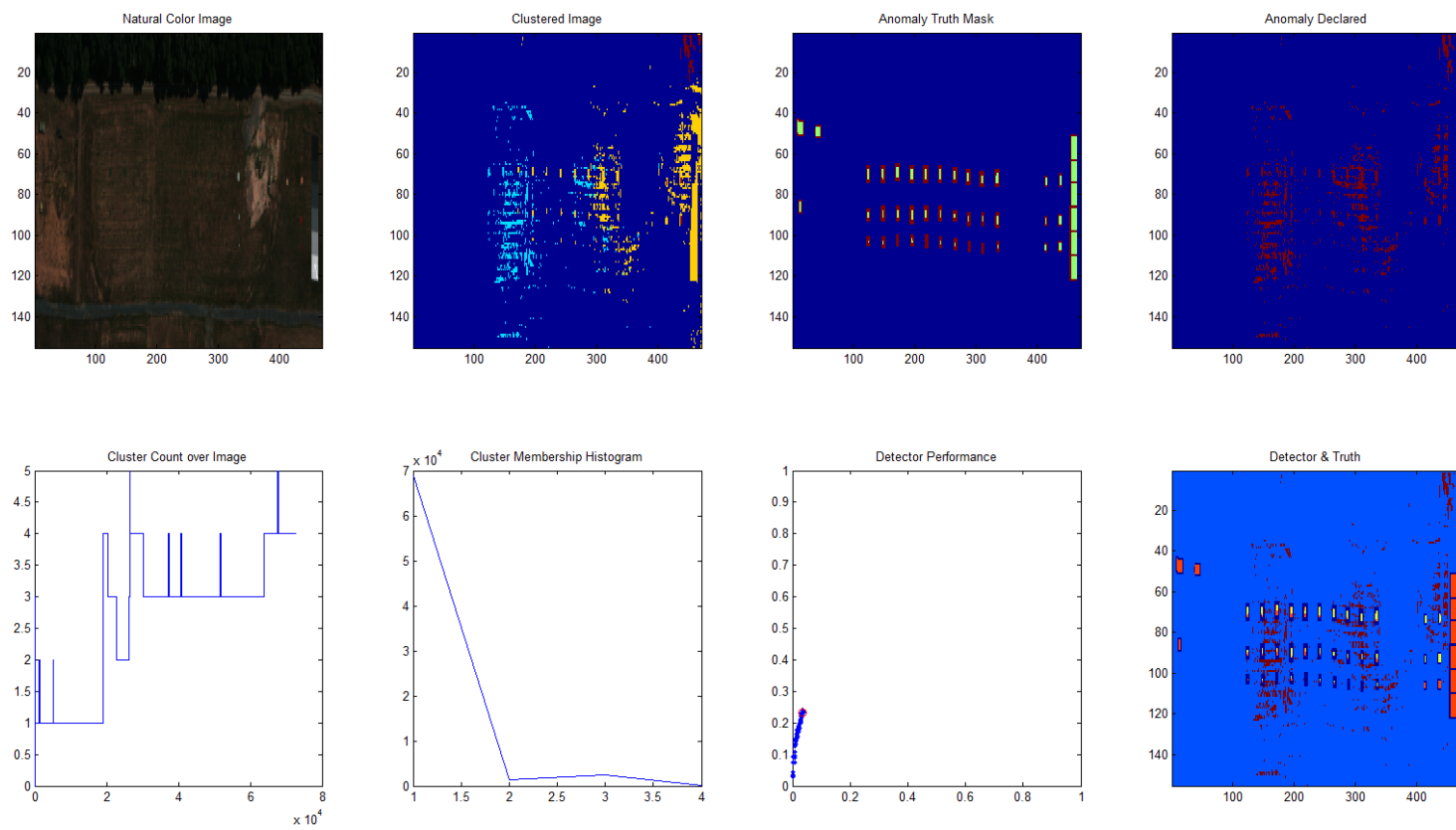


Figure 64: Image 15 Results (Table 5 Settings)

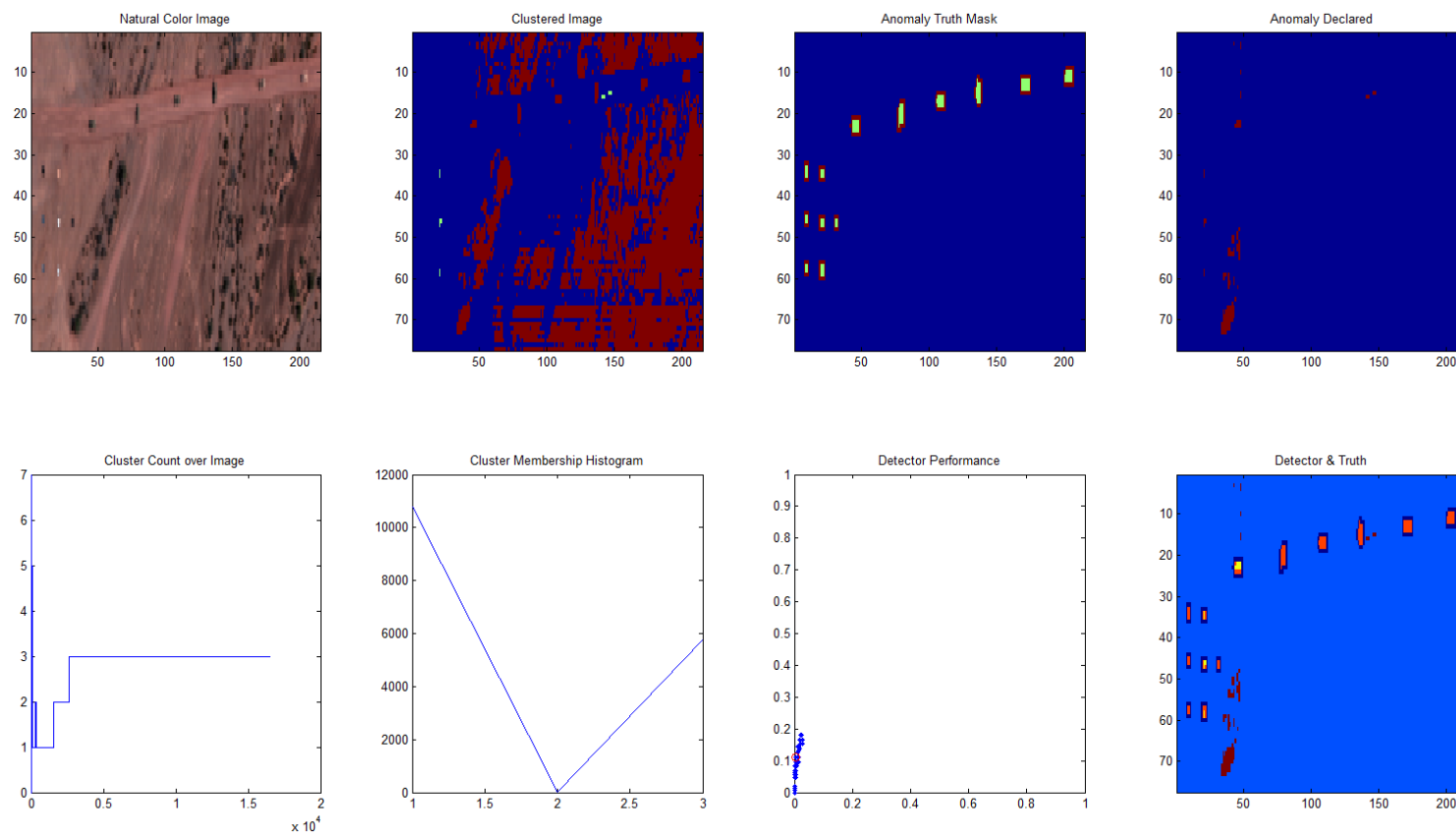


Figure 65: Image 16 Results (Table 5 Settings)

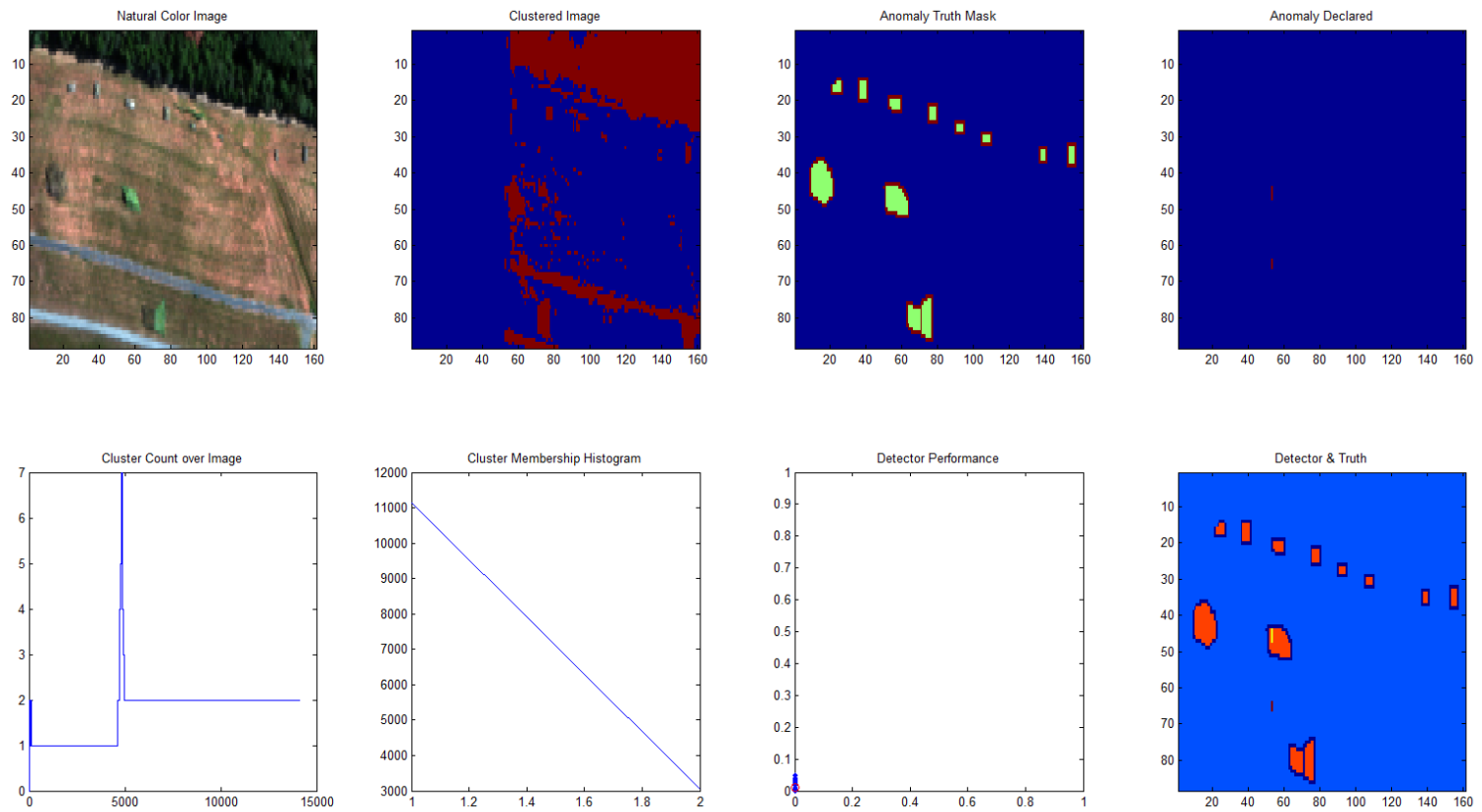


Figure 66: Image 17 Results (Table 5 Settings)

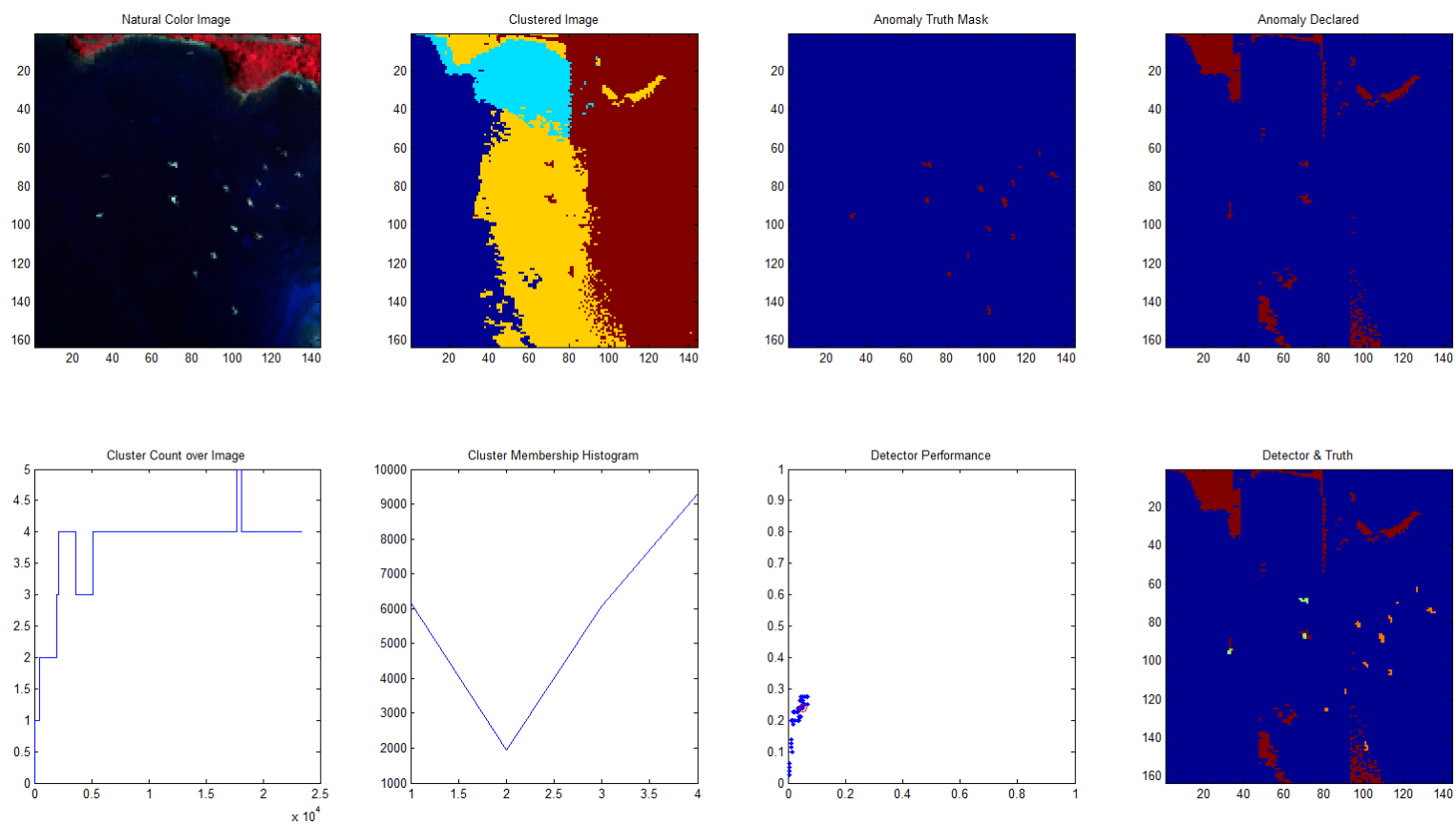


Figure 67: Image 18 Results (Table 5 Settings)

Appendix D

Image#	Ctot	C70	C80	C90	C95	C96	C97	C98	C99	EM Miss	Splits	Merges	Assign Time	Split Time	Merge Time	Total Time	TPR	FPR	TNR	FNR	Height	Width	Total Pixels	Actual # of Anomalies
1	31	1	1	2	2	3	3	5	18	20635	0	0	75.66	1.12	0.27	77.05	N/A	0.05	0.95	N/A	108	203	21924	0
2	58	2	2	2	3	3	3	3	18	52184	0	0	313.93	8.31	0.78	323.01	0.37	0.02	0.98	0.63	199	291	57909	1109
3	86	1	2	3	3	9	28	48	67	28308	0	0	277.73	2.67	0.43	280.82	0.60	0.10	0.90	0.40	160	191	30560	2953
4	39	2	3	3	3	3	3	8	23	20237	0	0	99.56	1.37	0.30	101.23	N/A	0.03	0.97	N/A	124	198	24552	0
5	77	1	1	1	1	1	11	25	39	19389	0	0	74.70	1.43	0.29	76.42	0.92	0.01	0.99	0.08	104	215	22360	4407
6	73	1	2	3	4	4	5	5	33	27697	0	0	198.81	6.07	0.64	205.32	0.89	0.08	0.92	0.11	152	312	47424	2749
7	25	1	1	2	2	2	2	2	6	19172	0	0	45.52	1.34	0.29	47.14	0.17	0.02	0.98	0.83	156	156	24336	748
8	71	1	1	1	20	27	34	41	48	9369	0	0	50.92	0.33	0.13	51.38	0.85	0.06	0.94	0.15	104	106	11024	381
9	77	1	16	44	58	60	63	66	69	3100	0	0	24.26	0.07	0.05	24.38	0.67	0.24	0.76	0.33	61	73	4453	175
10	96	2	2	3	3	8	26	45	64	28053	0	0	222.84	2.34	0.39	225.57	0.82	0.06	0.94	0.18	136	226	30736	773
11	58	1	2	2	2	2	2	2	22	33051	0	0	135.55	3.19	0.47	139.21	0.30	0.01	0.99	0.70	78	460	35880	3930
12	87	2	2	4	19	30	40	50	60	13118	0	0	97.46	0.69	0.20	98.35	0.61	0.10	0.90	0.39	80	205	16400	787
13	100	1	2	3	4	5	5	15	48	43304	0	0	326.92	7.72	0.75	335.38	0.81	0.09	0.91	0.19	150	355	53250	2667
14	49	1	2	2	16	22	28	34	40	8037	0	0	37.37	0.24	0.11	37.73	0.40	0.06	0.94	0.60	68	139	9452	825
15	122	1	1	2	3	3	4	9	53	58826	0	0	509.63	15.93	1.09	526.65	0.55	0.06	0.94	0.45	155	470	72850	4405
16	93	2	3	3	37	48	58	68	79	13294	0	0	135.90	0.72	0.21	136.82	0.42	0.10	0.90	0.58	77	215	16555	586
17	106	2	2	5	40	49	58	67	76	11898	0	0	95.71	0.57	0.18	96.45	0.88	0.16	0.84	0.13	88	161	14168	968
18	134	4	5	13	51	62	73	86	100	17556	0	0	143.04	1.29	0.29	144.63	0.99	0.13	0.87	0.01	163	144	23472	80

Figure 68: Summary of Clustering Performance (Table 6 Settings)

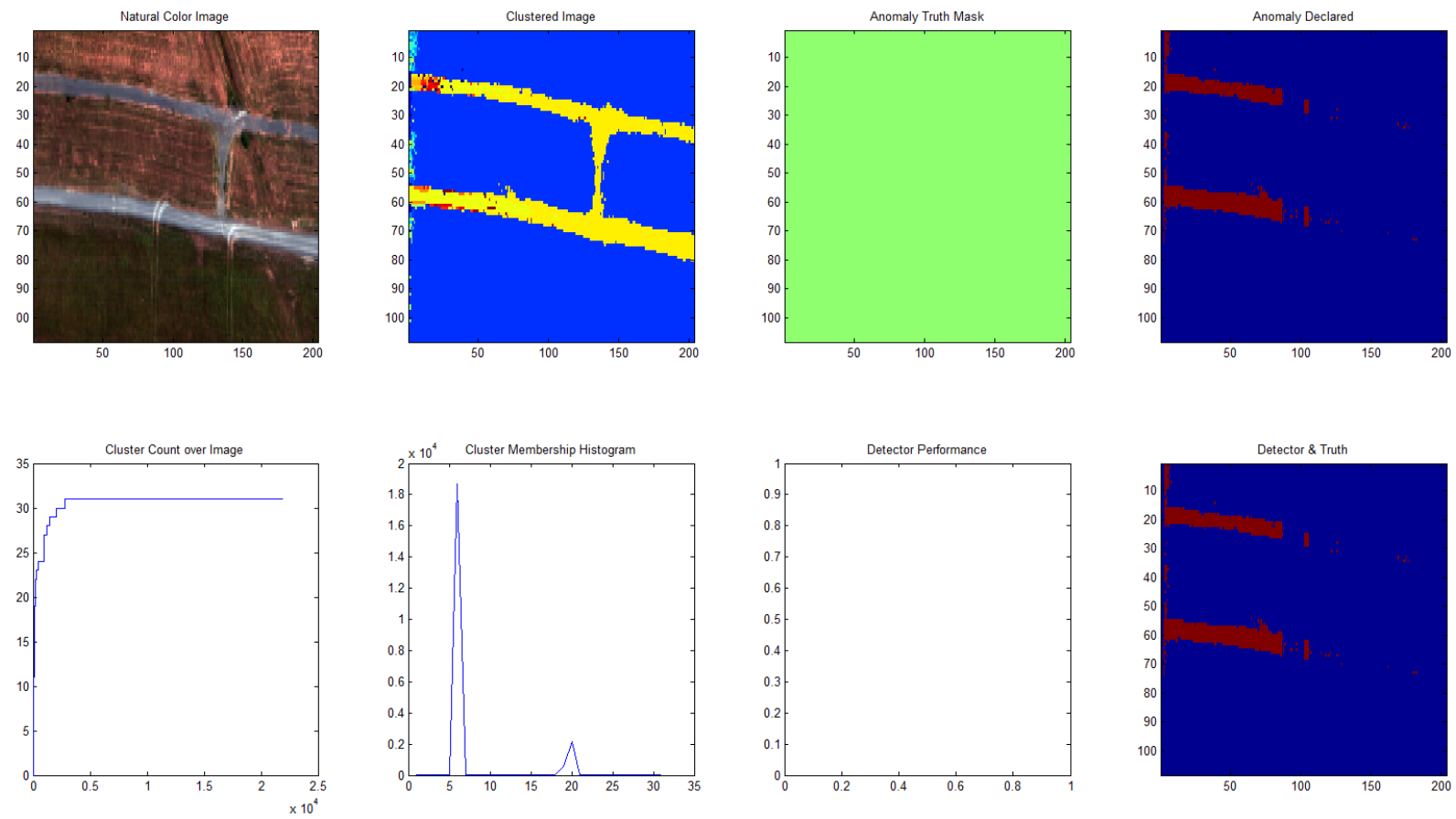


Figure 69: Image 1 Results (Table 6 Settings)

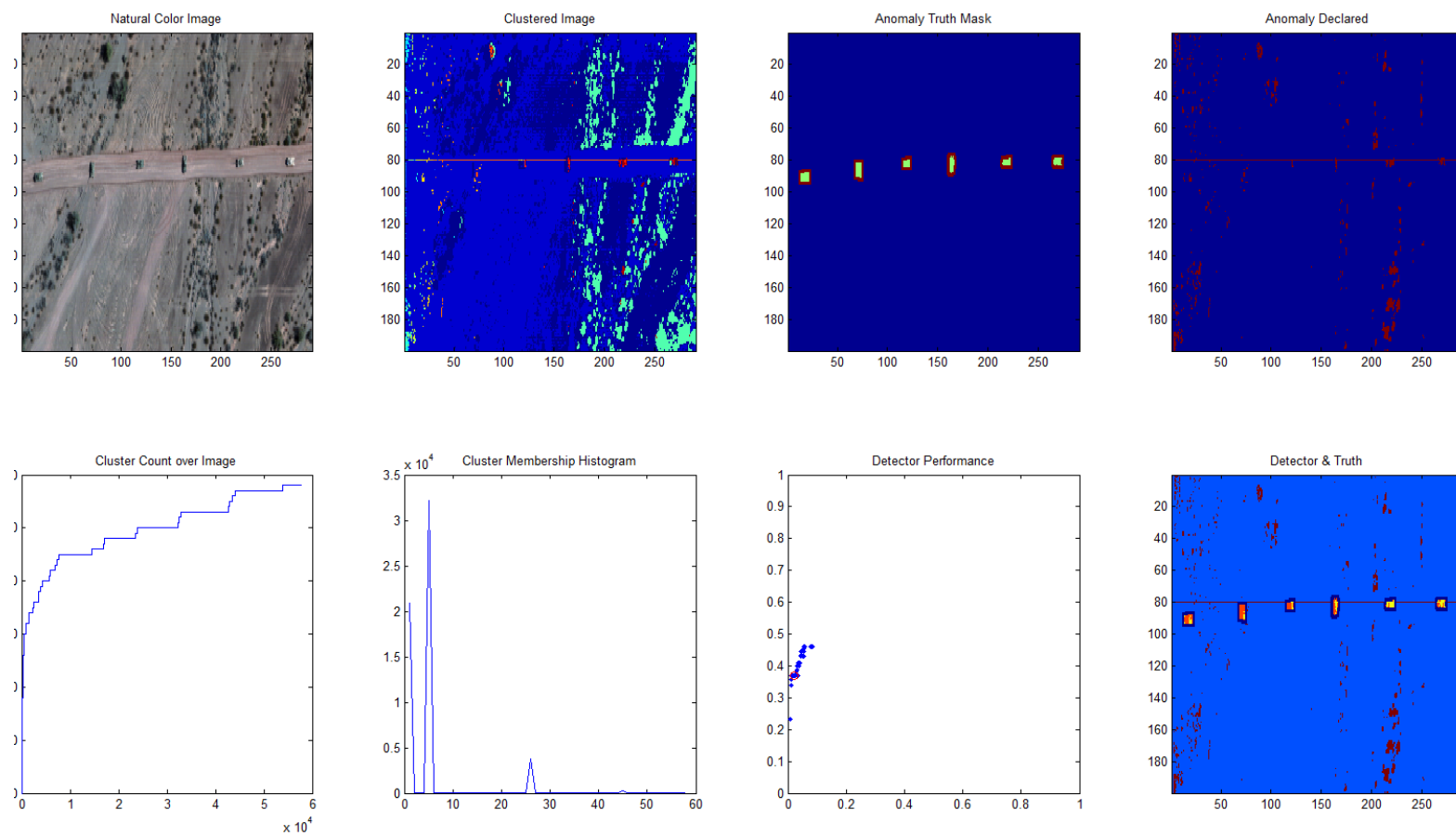


Figure 70: Image 2 Results (Table 6 Settings)

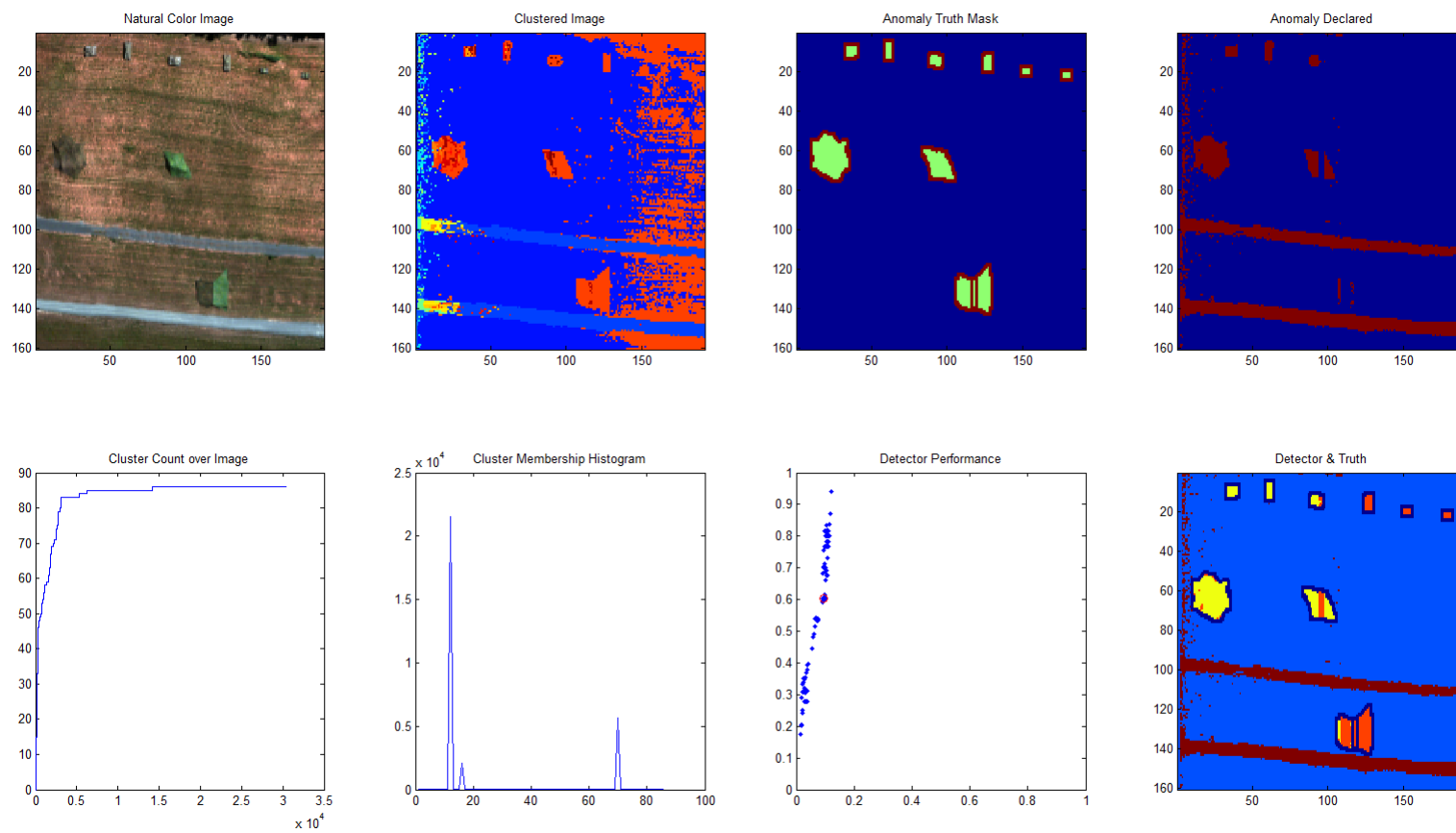


Figure 71: Image 3 Results (Table 6 Settings)

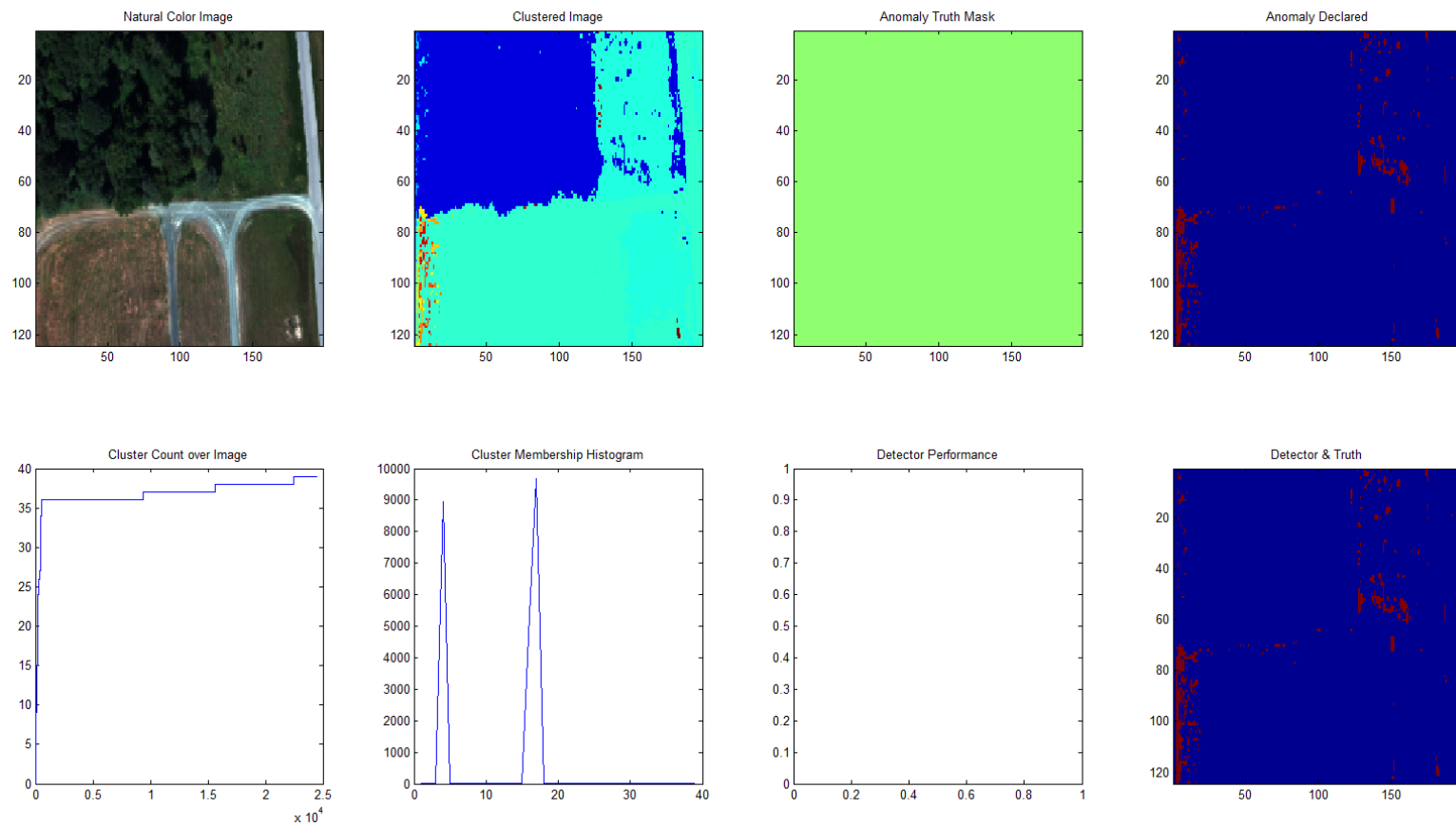


Figure 72: Image 4 Results (Table 6 Settings)

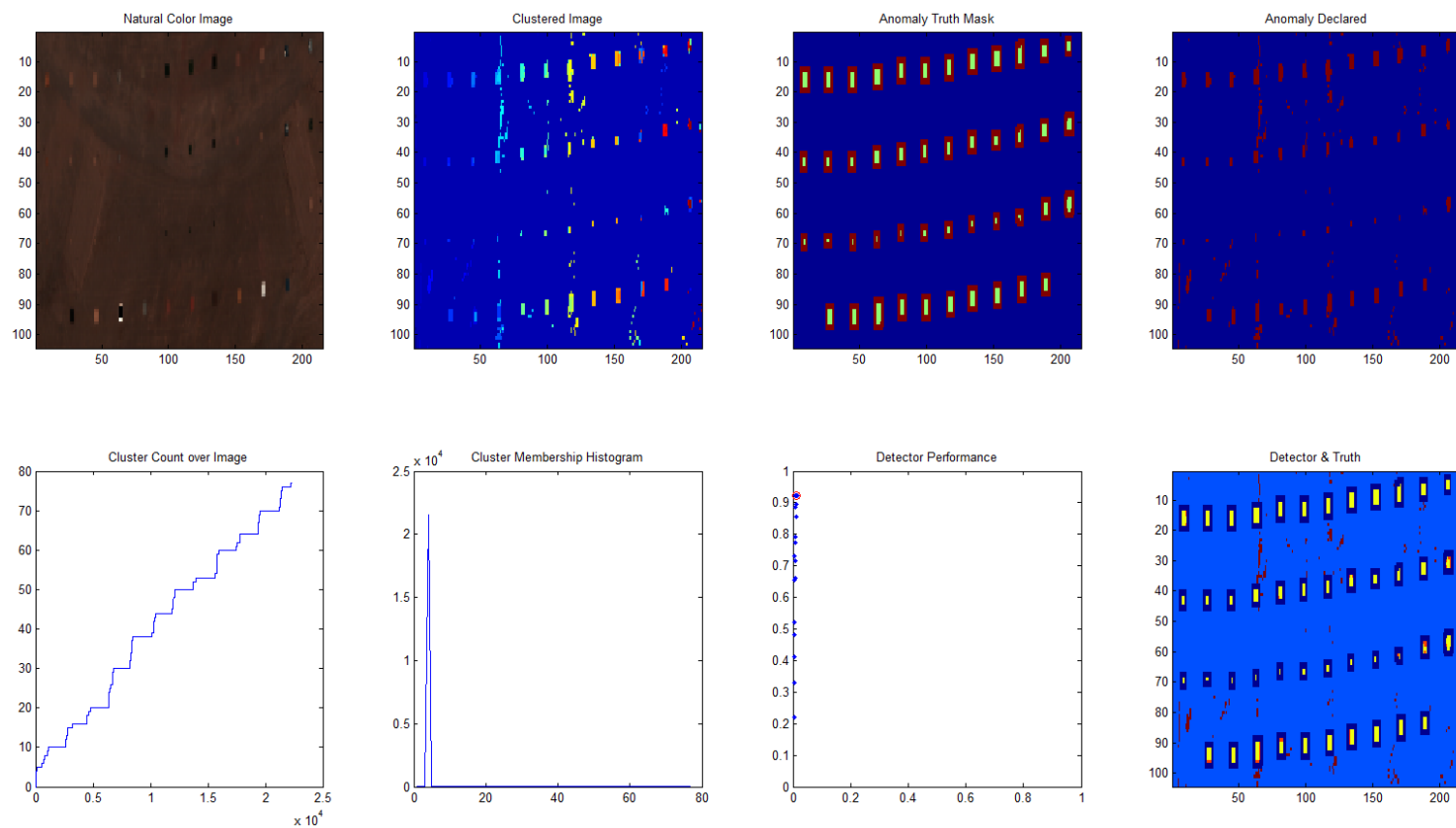


Figure 73: Image 5 Results (Table 6 Settings)

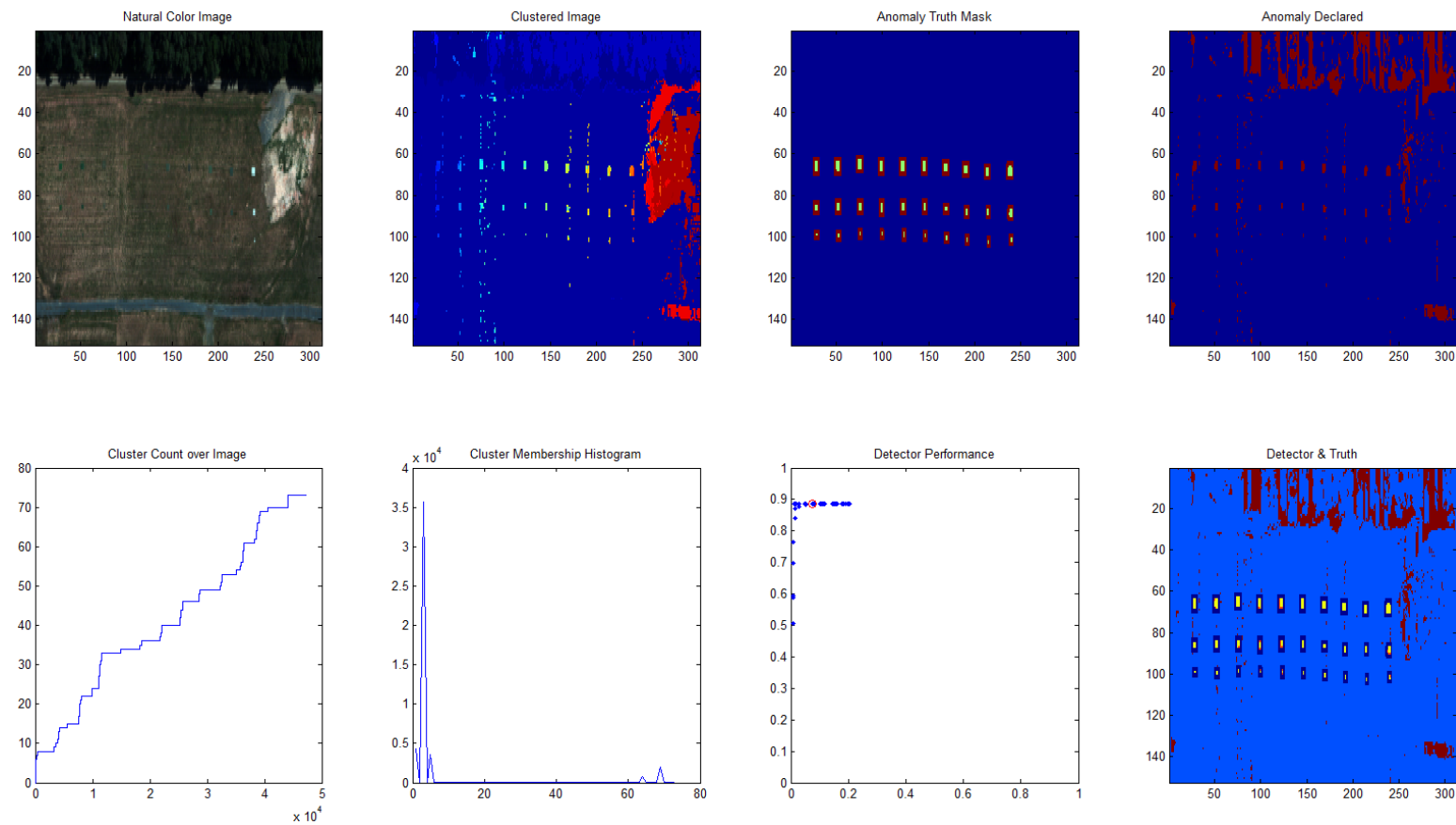


Figure 74: Image 6 Results (Table 6 Settings)

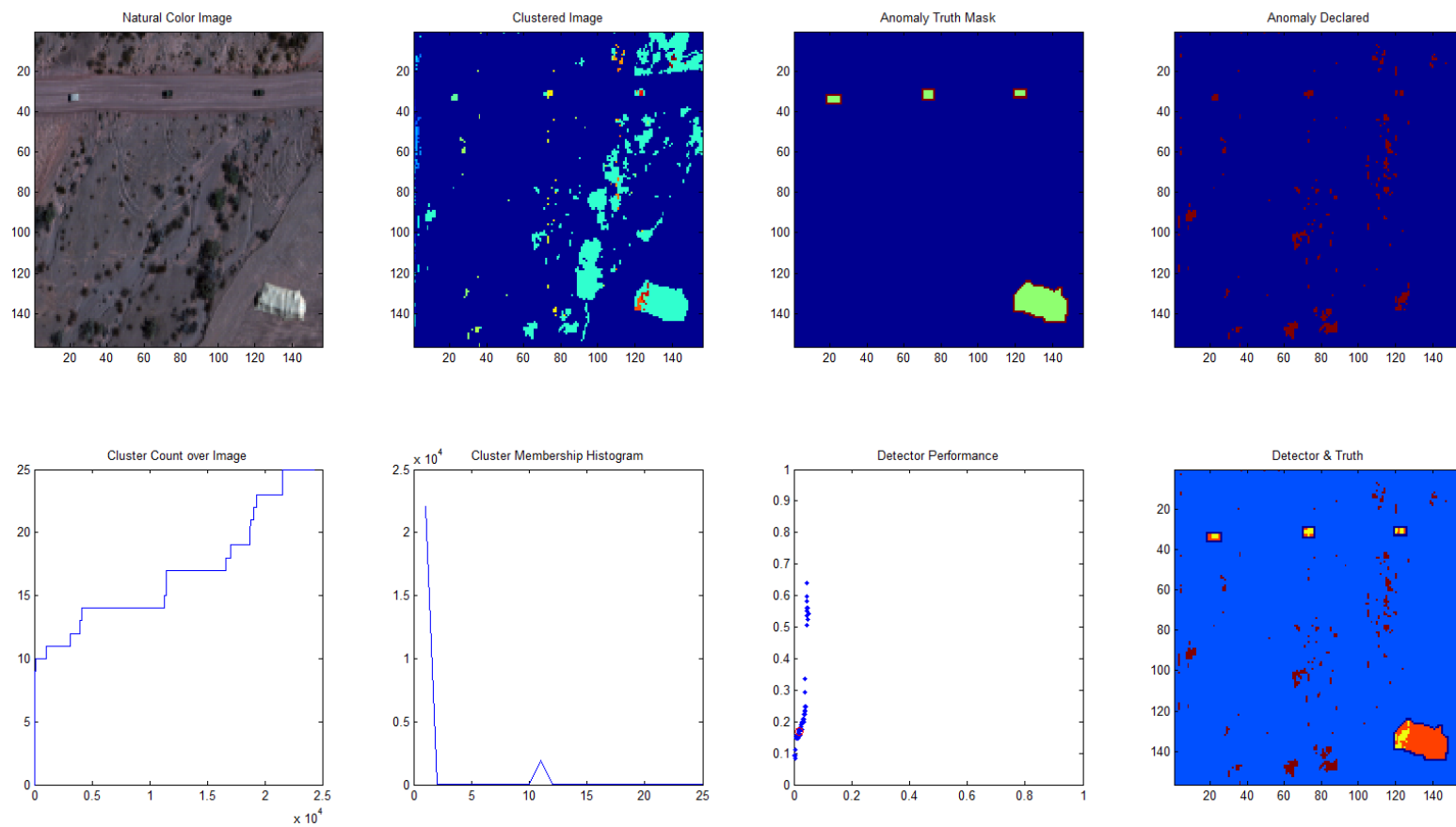


Figure 75: Image 7 Results (Table 6 Settings)

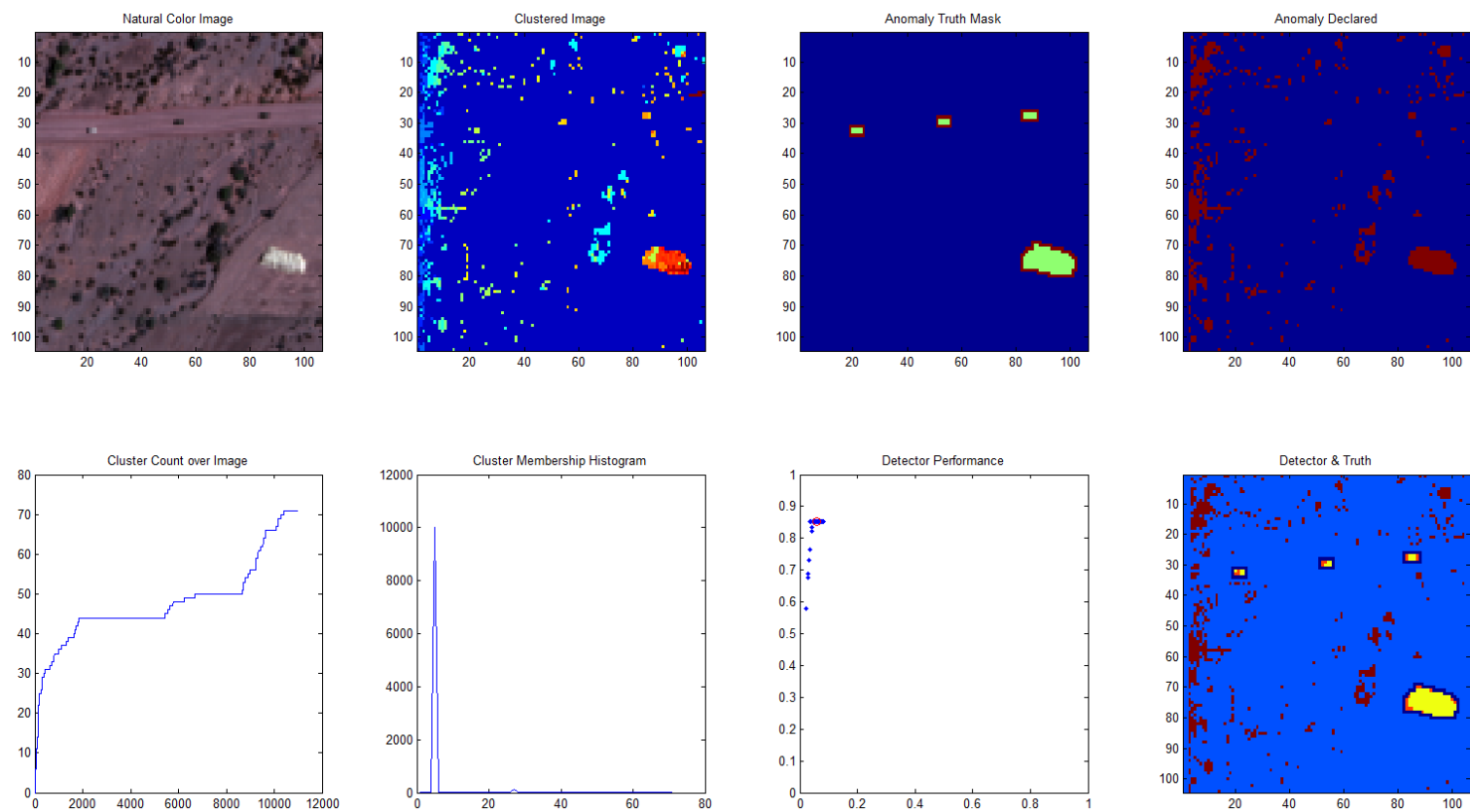


Figure 76: Image 8 Results (Table 6 Settings)

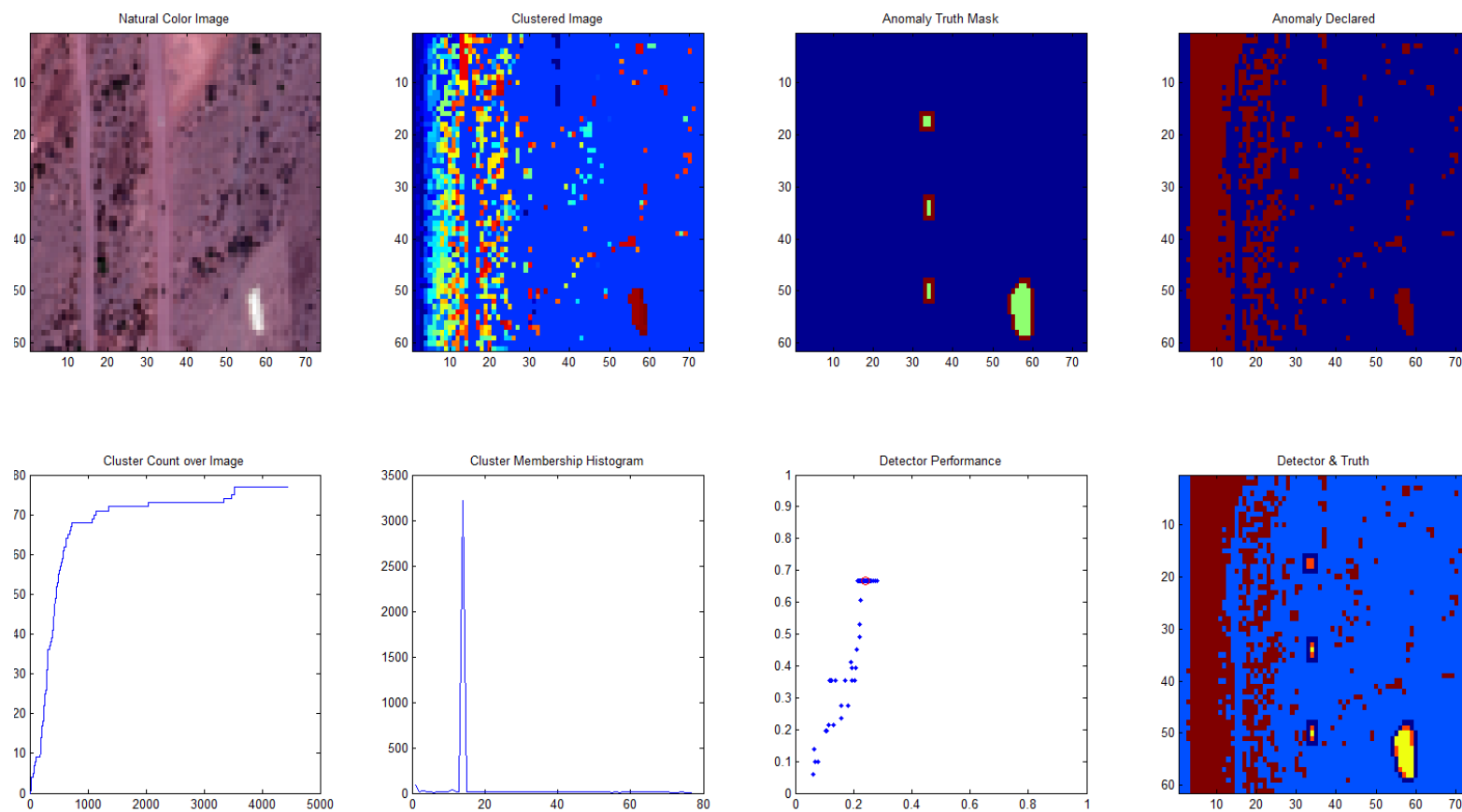


Figure 77: Image 9 Results (Table 6 Settings)

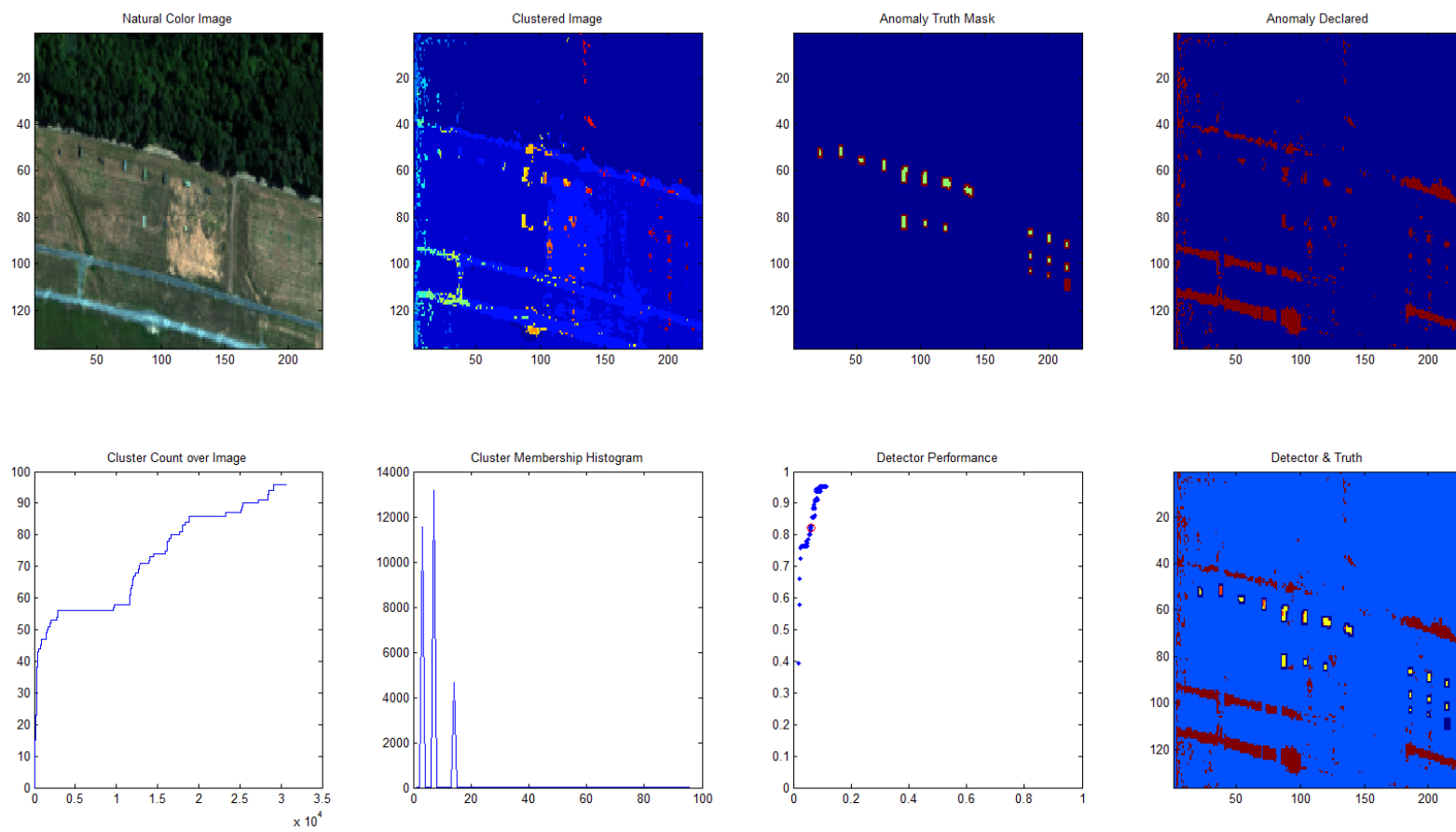


Figure 78: Image 10 Results (Table 6 Settings)

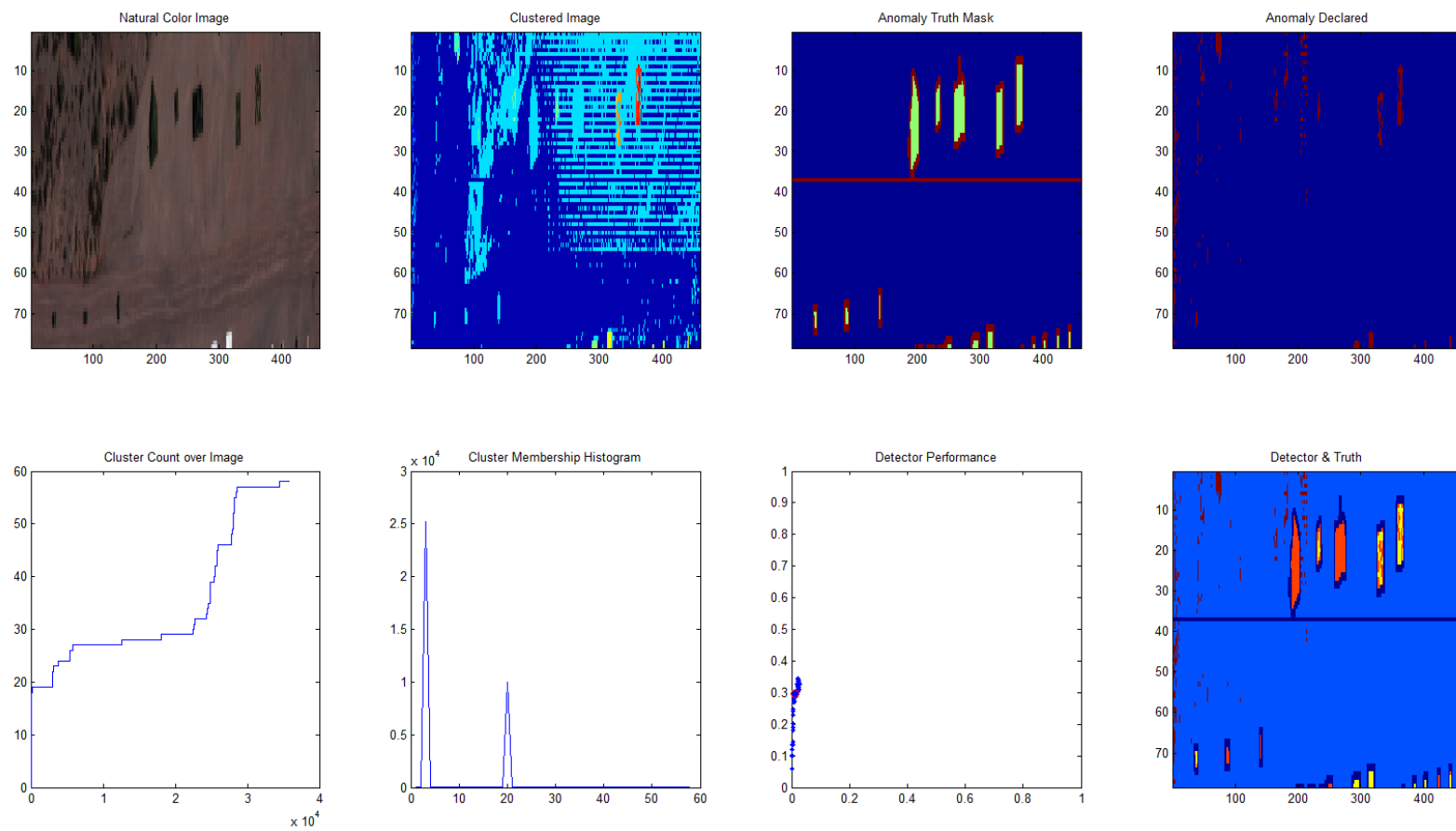


Figure 79: Image 11 Results (Table 6 Settings)

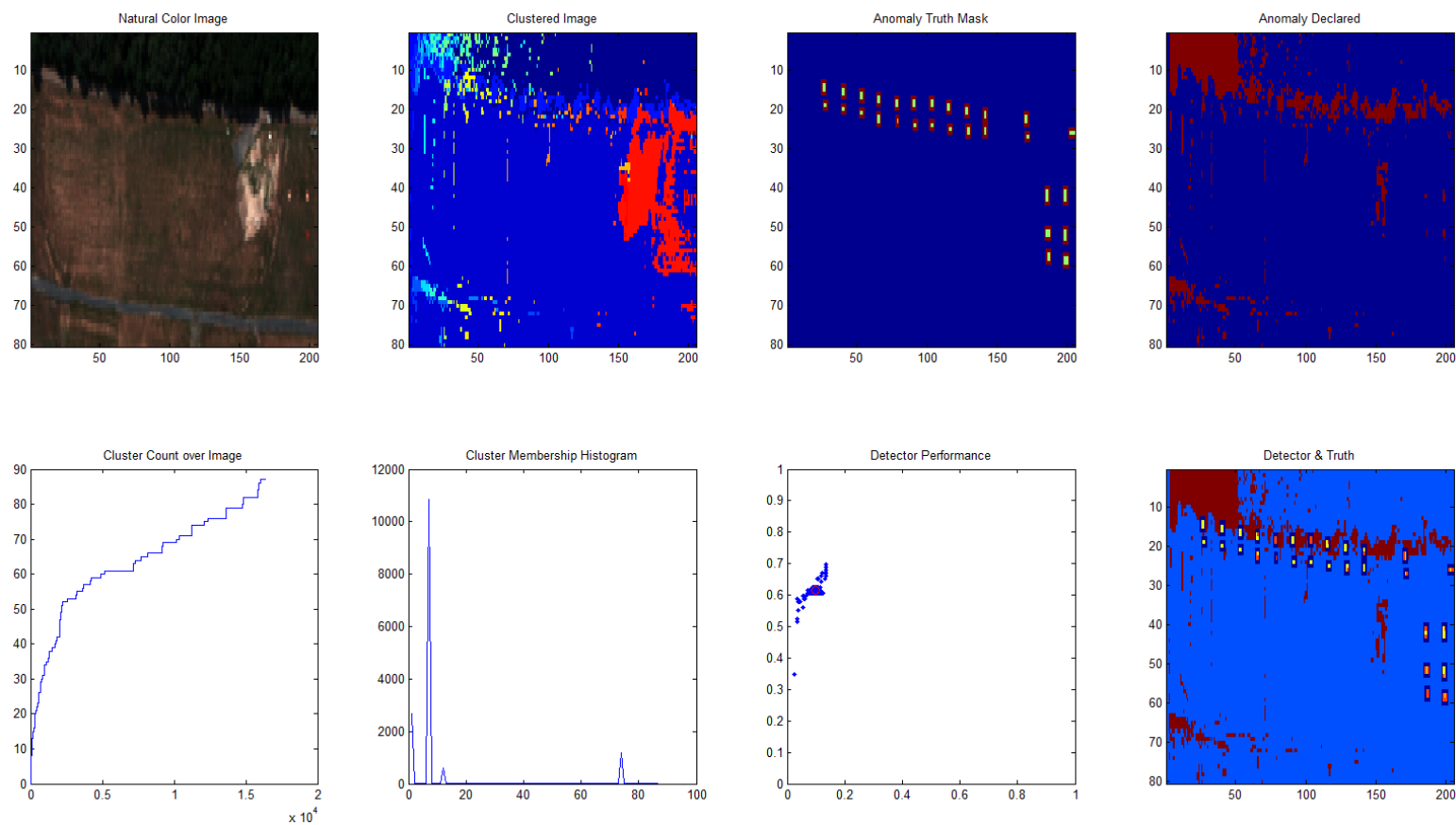


Figure 80: Image 12 Results (Table 6 Settings)

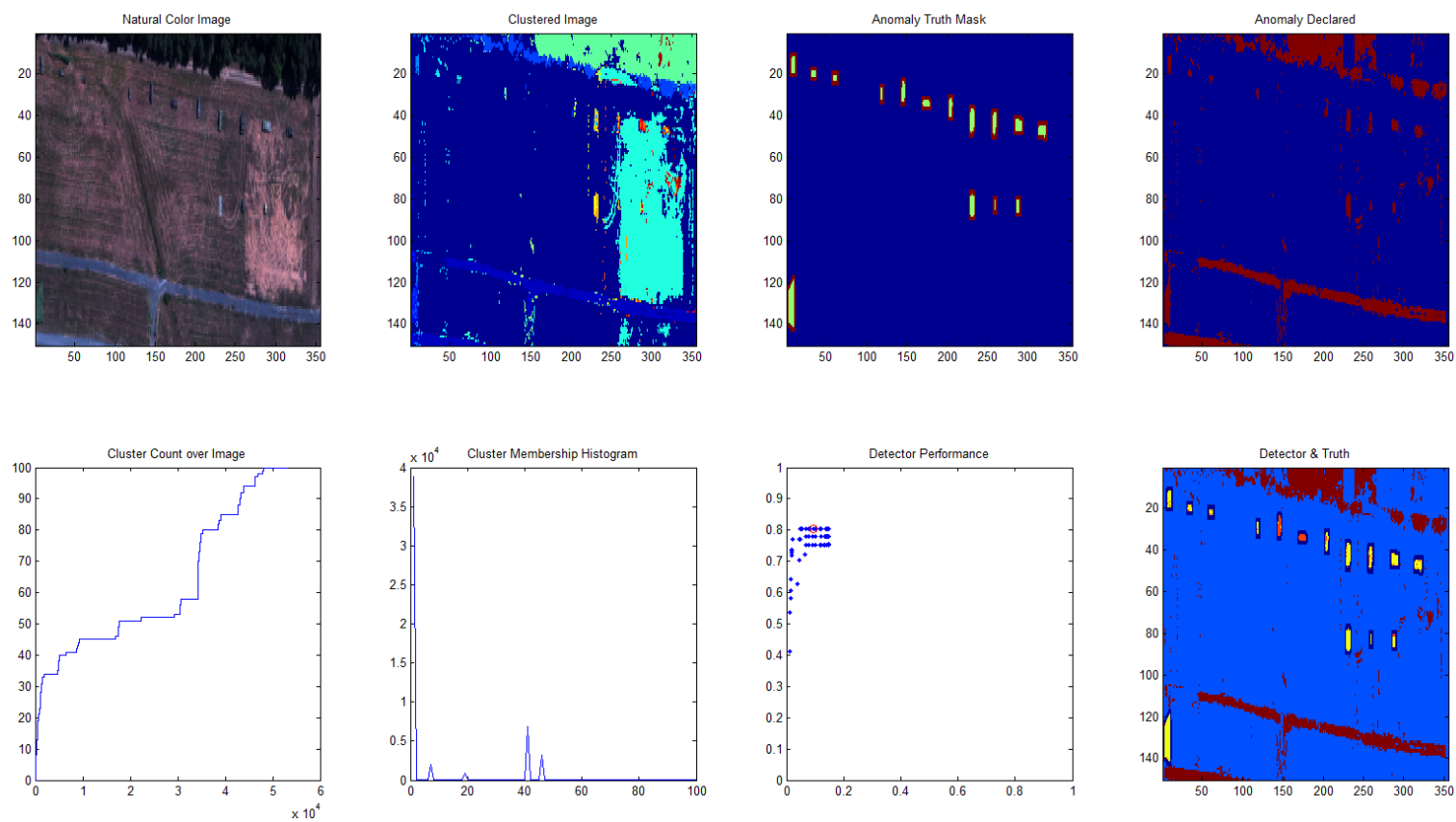


Figure 81: Image 13 Results (Table 6 Settings)

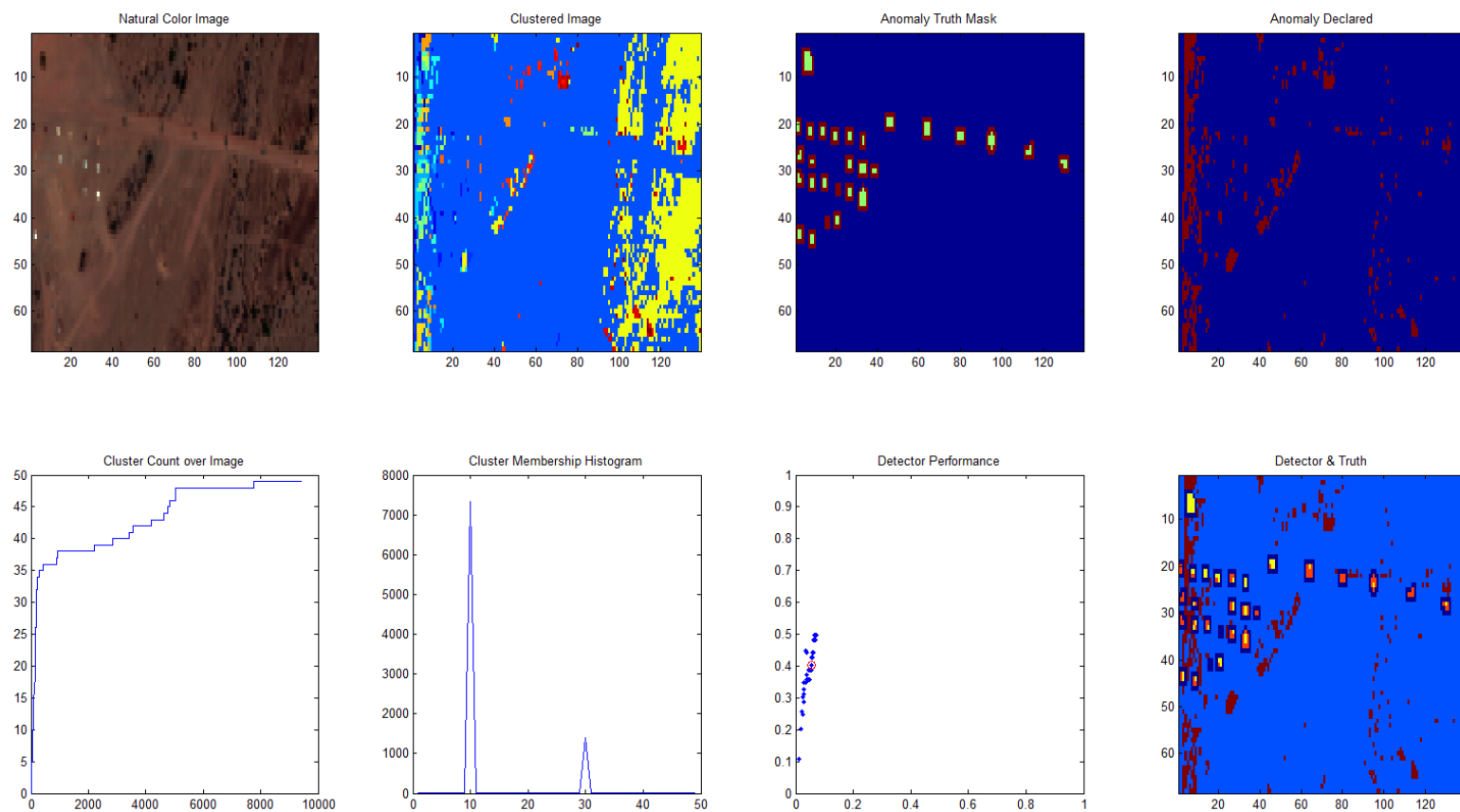


Figure 82: Image 14 Results (Table 6 Settings)

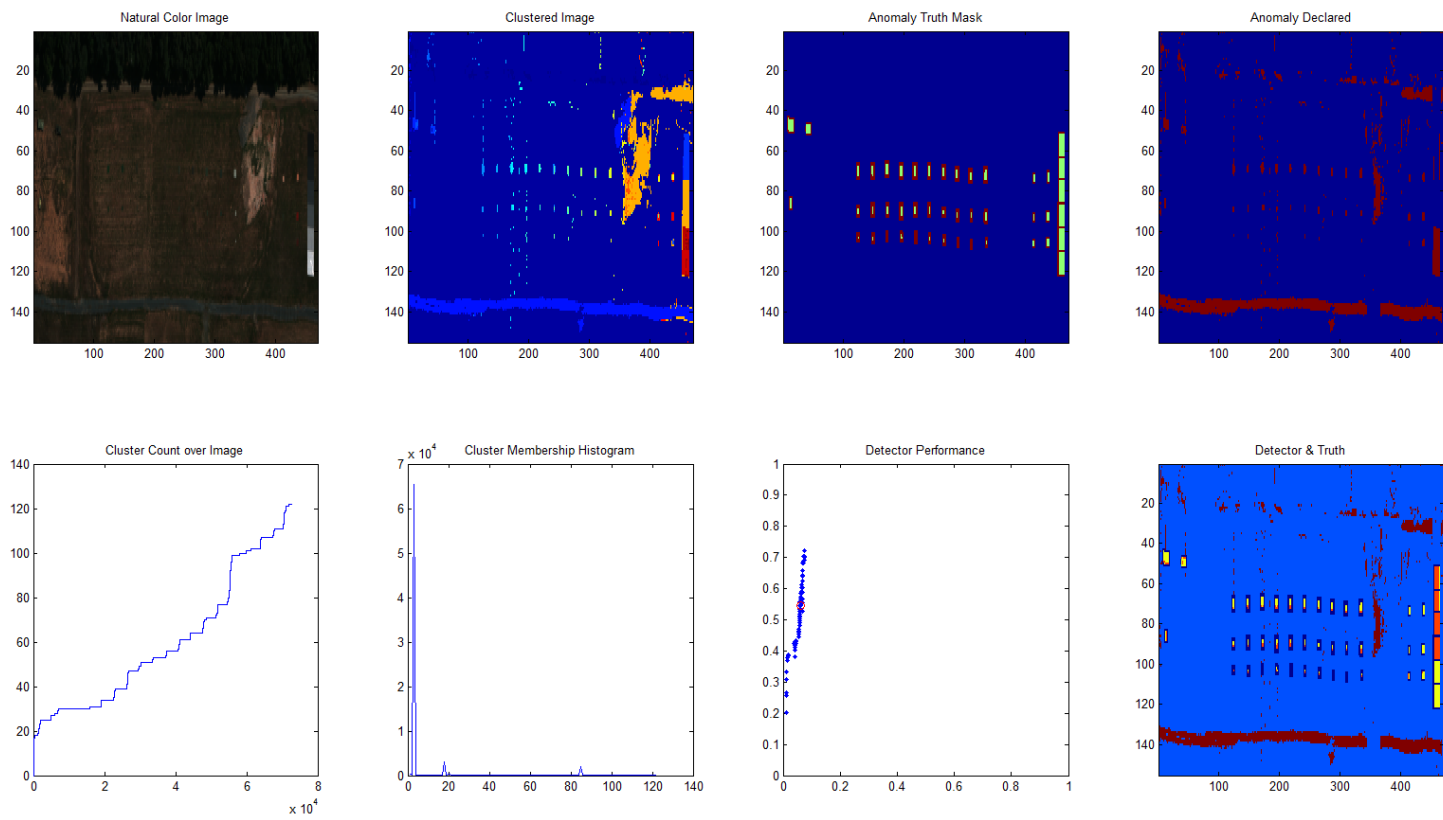


Figure 83: Image 15 Results (Table 6 Settings)

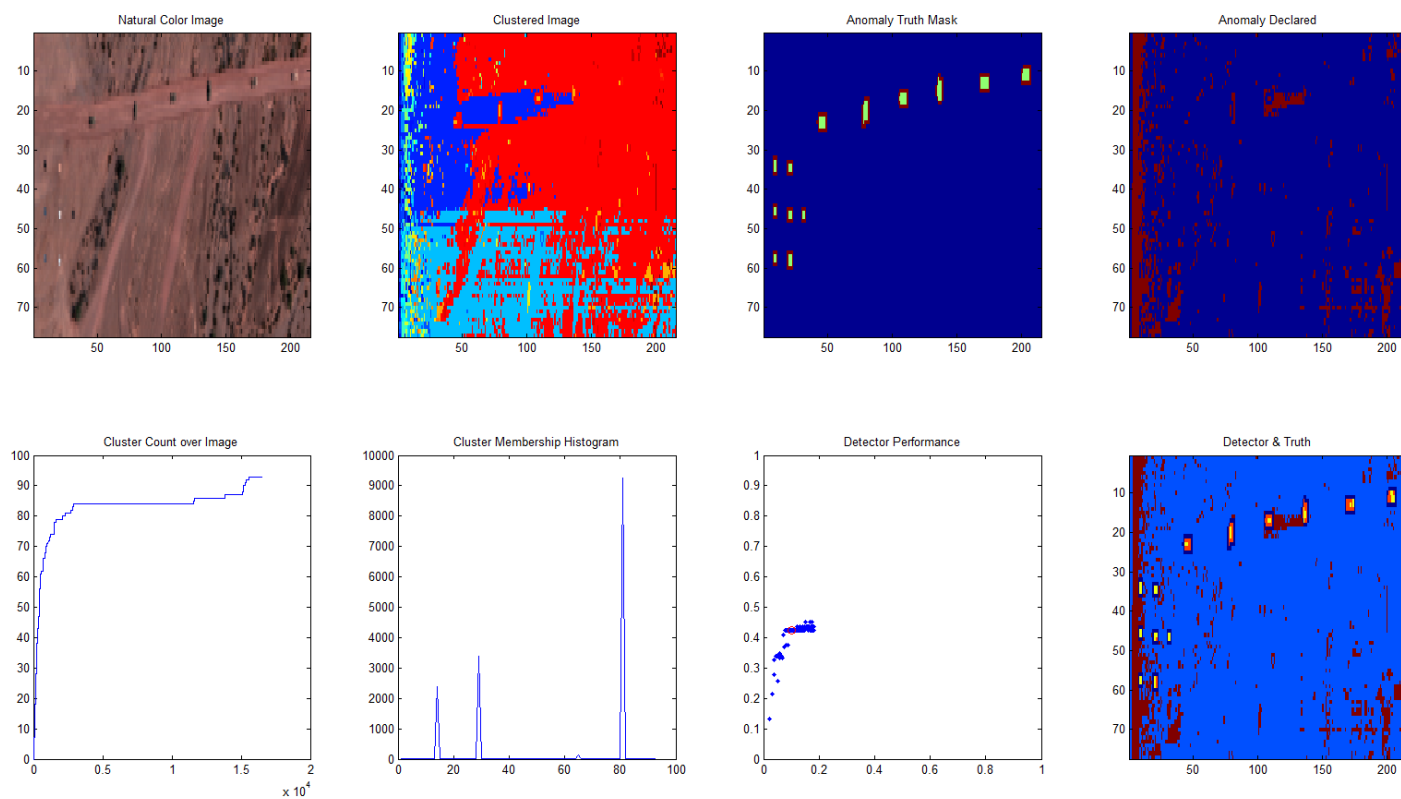


Figure 84: Image 16 Results (Table 6Table 6Table 5 Settings)

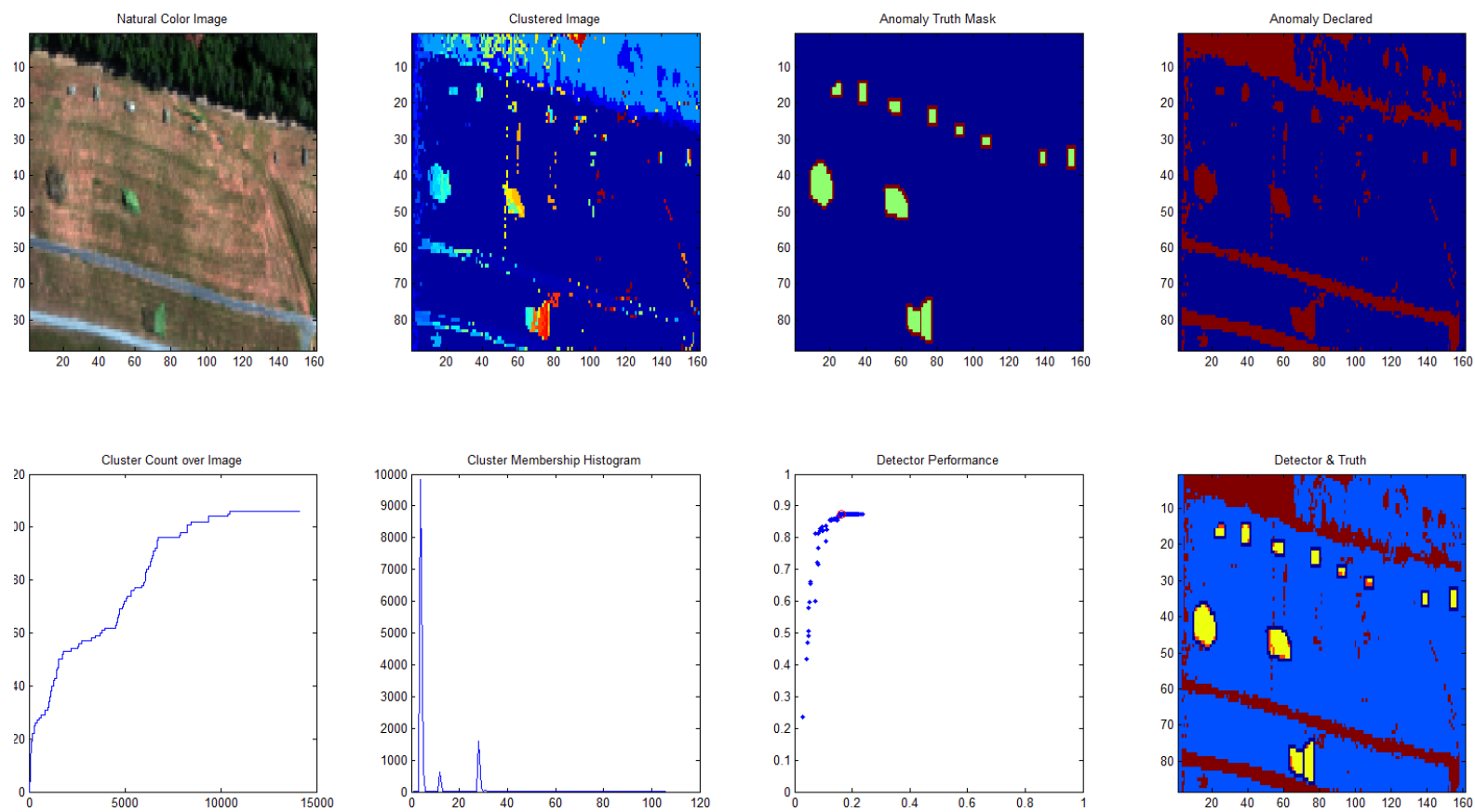


Figure 85: Image 17 Results (Table 6 Settings)

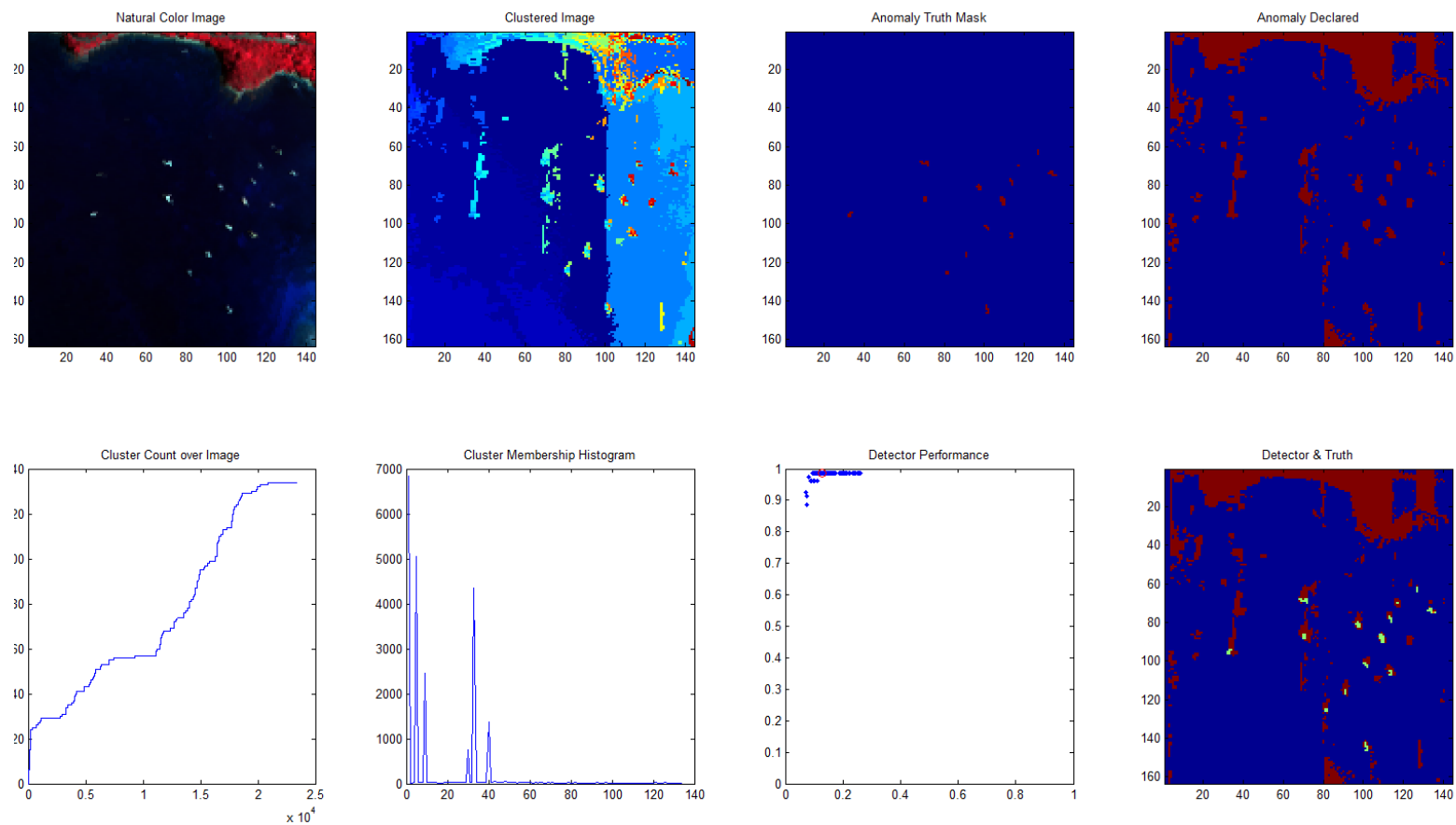


Figure 86: Image 18 Results (Table 6 Settings)

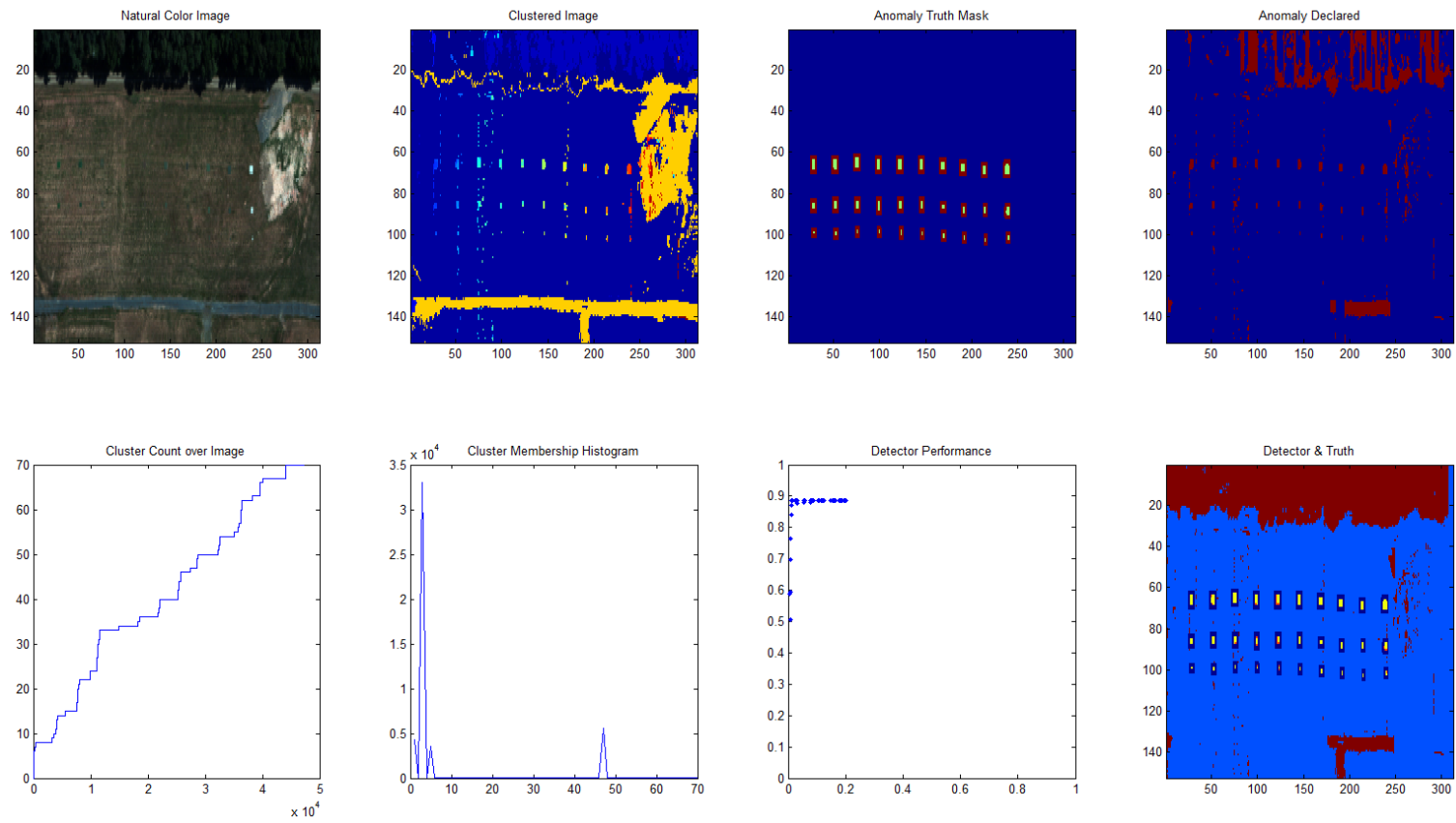


Figure 87: Image 6 Results with Split Cluster Enabled (Table 7 Settings)

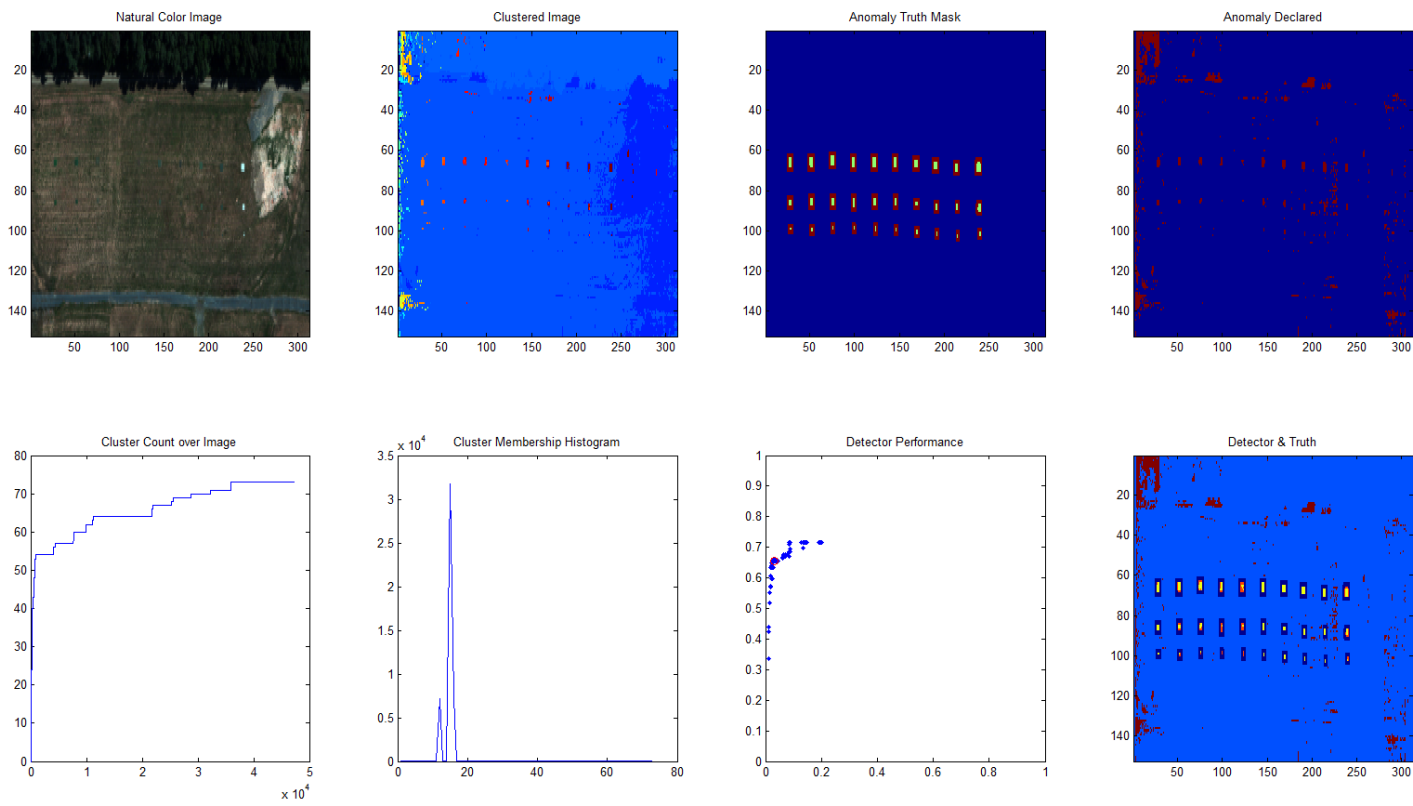


Figure 88: Image 6 Results with Sand Generated Principal Components (Table 6 Settings)



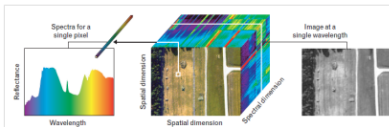
Online Cluster Analysis Supporting Real-Time Anomaly Detection in Hyperspectral Imagery



RESEARCH OBJECTIVE:

- Implement an online clustering algorithm that can support real time anomaly detection

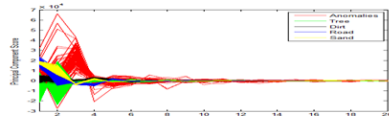
Spectral Signature Image Cube Image of One Spectral Band



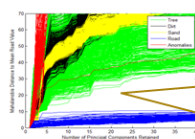
Hyperspectral Images contain hundreds of correlated bands. Non-natural objects have distinct signatures, which allow them to be detected in hyperspectral images. This detection can be enhanced or performed by clustering, but online detection methods require online clustering to benefit from this enhancement.

SUBSPACE CHARACTERIZATION

Hyperspectral image processing requires the use of a subspace to deal with the high correlation between bands. Principal component analysis (PCA) is often used, and exhibits features that allow for online clustering using Mahalanobis distances



$$D_{Mahalanobis} = \sqrt{(x - \bar{x})^T S^{-1} (x - \bar{x})}$$

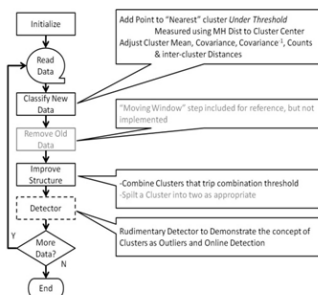


Consistent gap in Mahalanobis distance between similar and different signatures allows for a constant set of parameters for online clustering



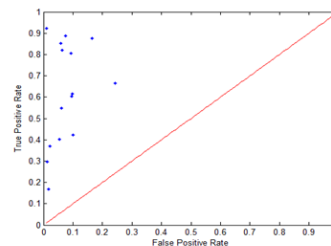
Maj Elwood T. Waddell
Department of Operational Sciences (ENS)
Advisor: Dr. Kenneth Bauer
Sponsor: AFRL/RY

ALGORITHM

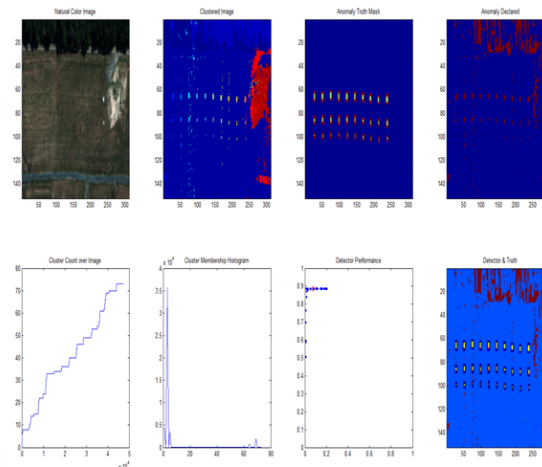


DETECTOR SUMMARY RESULTS

A rudimentary detector based on cluster counts showed promising results across 15 images

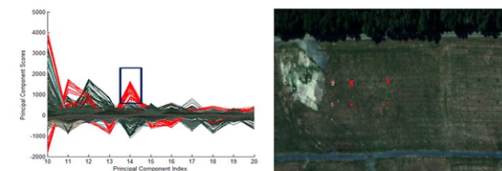


EXAMPLE CLUSTERING RESULTS



OTHER RESULTS

Anomalies in the Principal Component Space appear to uniquely register separately from the background on a component by component basis, potentially forming the basis for an ensemble detection method or GUI for analyst in the individual higher dimensional principal components.



Bibliography

- Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). A Framework for Clustering Evolving Data Streams. *Proceedings of the 29th VLDB Conference*. Berlin, Germany: Morgan Kaufmann .
- Ball, G. H., & Hall, D. J. (1965). *ISODATA, A Novel Method of Data Analysis and Pattern Classification*. Menlo Park, California: Stanford Research Institute.
- Biatov, K. (2010). A fast speaker indexing using vector quantization and second order statistics with adaptive threshold computation. *11th Annual Conference of the International Speech Communication Association 2010 (INTERSPEECH 2010)* (pp. 1453-1456). Makuhari, Japan: Curran Associates, Inc.
- Bigley, A. (2013). *Horn's Curve Estimation Through Multidimensional Interpolation*. Wright Patterson Air Force Base OH: MS thesis, AFIT-ENS-13-M-01. Air Force Institute of Technology.
- Bihl, T. J., Bauer, K. W., & Friend, M. A. (2013). Selecting Principal Components using Index Screening for Hyperspectral Anomaly Detection. *Air Force Institute of Technology White Paper* .
- Bush, K. R. (2012). *Using QR Factorization for Real-Time Anomaly Detection of Hyperspectral Images*. Wright Patterson Air Force Base OH: MS thesis, AFIT-OR-MS-ENS-12-04. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).
- Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence* , 678-698.
- Carlotto, M. J. (2005). A Cluster-Based Approach for Detecting Manmade Objects and Changes in Imagery. *IEEE Transactions on Geoscience and Remote Sensing* , 374-387.
- Chakrabarti, D., Kumar, R., & Tomkins, A. (2006). Evolutionary Clustering. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 554-560). Philadelphia, PA: ACM.
- Chakrabarti, K., & Mehrotra, S. (2000). Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Spaces. *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)* (pp. 89-100). Cairo, Egypt: Morgan Kaufmann Publishers Inc.

- Chakraborty, S., & Nagwani, N. K. (2011). Analysis and Study of Incremental DBSCAN Clustering Algorithm. *International Journal of Enterprise Computing and Business Systems* .
- Chakraborty, S., Kagwani, N. K., & Dey, L. (2011). Performance Comparison of Incremental K-means and Incremental DBSCAN Algorithms. *International Journal of Computer Applications* , 14-18.
- Cobb, C. M. (1988, May 27). A Quantitative Comparison of an ISODATA and a Single-Pass Clustering Algorithm Applied over Simulated Multispectral Image Data Sets. Kentucky: Murray State University.
- Cormen, T. H., Leiserson, C. E., & Rivest, R. L. (1998). *Introduction to Algorithms*. Cambridge, Massachusetts: MIT Press and McGraw-Hill.
- Dillon, W. R., & Goldstein, M. (1984). *Multivariate Analysis*. New York: John Wiley & Sons.
- Ding, M., Tian, Z., & Xu, H. (2010). Adaptive kernel principal component analysis. *Signal Processing* , 1542-1553.
- Dinh, V. C., Leitner, R., Paclik, P., & Duin, R. P. (2009). A Clustering Based Method for Edge Detection in Hyperspectral Images. *SCIA '09 Proceedings of the 16th Scandinavian Conference on Image Analysis* (pp. 580-587). Oslo, Norway: Springer-Verlag.
- Donohue, J. (1991). *Introductory Review of Target Discrimination Criteria*. Hanscom Air Force Base, Massachusetts: Phillips Laboratory.
- Duan, L., Xu, L., Liu, Y., & Lee, J. (2009). Cluster-based outlier detection. *Annals of Operations Research* , 151-168.
- Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification* (2nd ed.). New York: John Wiley & Sons, Inc.
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (96). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)* (pp. 226-231). Portland, Oregon: AAAI.

- Goovaerts, P., Jacquez, G. M., & Marcus, A. (2005). Geostatistical and Local Cluster Analysis of High Resolution Hyperspectral Imagry for Detection of Anomalies. *Remote Sensing of Environment* , 351-367.
- Han, X., & Zhao, T. (2005). Auto-K Dynamic Clustering Algorithm. *Journal of Animal and Veterinary Advances* , 535-539.
- Hardin, J., & Rocke, D. M. (2004). Outlier Detection in the Multiple Cluster Setting Using the Minimum Covariance Determinant Estimator. *Computational Statistics and Data Analysis* , 625-638.
- Hobbs, P. C. (2000). *Building Electro-Optical Systems, Making It All Work*. New York: John Wiley & Sons, Inc.
- Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition Letters* 31 , 651-666.
- Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data Clustering: A Review. *ACM Computing Surveys* , 264-323.
- Jain, A. K., Topchy, A., Law, M. H., & Buhmann, J. M. (2004). Landscape of Clustering Algorithms. *Proceedings of the 17th international Conference on Pattern Recognition (ICPR '04)* (pp. 260-263). Cambridge, England: IEEE.
- Johnson, R. J. (2008). *Improved Feature Extraction, Feature Selection, and Identification Techniques that Create a Fast Unsupervised Hyperspectral Target Detection Algorithm*. Wright Patterson Air Force Base OH: MS thesis, AFIT/GOR/ENS/08-07. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).
- Kass, R. E., & Wasserman, L. (1995). A Reference Bayesian Test for Nested Hypotheses and It's Relationship to the Schwartz Criterion. *Journal of the American Statistical Association* , 928-934.
- Koschan, A., & Abidi, M. (2005). Detection and Classification of Edges in Color Images, *Signal Processing Magazine*, Special Issue on Color Image Processing, Vol. 22, No. 1. pp. 64-73.
- Kriegel, H.-P., Kroeger, P., & Zimek, A. (2009). Clustering High-Dimensional Data: A Survery on Subspace Clustering, Pattern-Based Clustering, and Correlation Clustering. *ACM Transactions on Knowledge Discovery from Data* , 1:1-1:58.

- Kutner, M. H., Nachstheim, C. J., Neter, J., & Li, W. (2005). *Applied Linear Statistical Models, 5th Edition*. Boston: McGraw-Hill Irwin.
- Li, J., Huang, X., Selke, C., & Yong, J. (2007). A fast algorithm for finding correlation clusters in noise data. *Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)* (pp. 639-647). Nanjing, China: Springer-Verlag Berlin, Heidelberg.
- Li, W., Yue, H. H., Valle-Cervantes, s., & Qin, S. J. (2000). Recursive PCA for adaptive process monitoring. *Journal of Process Control* , 471-486.
- Liu, X., Kruger, U., Littler, T., Xie, L., & Want, S. (2009). Moving window kernel PCA for adaptive monitoring of nonlinear processes. *Chemometrics and Intelligent Laboratory Systems* , 132-143.
- Looney, C. G. (1997). *Pattern Recognition Using Neural Networks*. Oxford, UK: Oxford University Press.
- Mathworks, T. (2012). MATLAB: Version 7.14.0.739 (R2012a).
- Matteoli, S., Diani, M., & Corsini, G. (2010, July). A Tutorial Overview of Anomaly Detection in Hyperspectral Images. *IEEE A&E Systems Magazine* , pp. 5-27.
- Messer, A. J. (2011). *Contextual Detection of Anomalies within Hyperspectral Images*. Wright Patterson Air Force Base OH: MS Thesis, AFIT-OR-MS-ENS-11-15. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).
- Mok, P. Y., Huang, H. Q., & Kwok, J. S. (2012). A robust adaptive clustering analysis method for automatic identification of clusters. *Pattern Recognition* , 3017-3033.
- O'Callaghan, L., Mishra, N., Meyerson, A., Guha, S., & Motwani, R. (2002). Streaming-Data Algorithms for High-Quality Clustering. *18th International Conference on Data Engineering* (pp. 685-694). San Jose, CA: IEEE.
- Pelleg, D., & Moore, A. (2000). X-means: Extended K-means with Efficient Estimation of the Number of Clusters. *Proceedings of the 17th International Conference on Machine Learning*, (pp. 727-734).
- Reed, I. S., & Yu, X. (1990). Adaptive Multiple-Band CFAR Detection of an Optical Pattern with Unknown Spectral Distribution. *IEEE Transactions on Acoustics, Speech, and Signal Processing* , 1760-1770.

- Rickard, L. J., Basedow, R., Zalewski, E., Silverglate, E., & Silverglate, P. (1993). HYDICE: An Airborne System for Hyperspectral Imaging". *Proceedings of the SPIE* , 173-179.
- Robinson, G. S. (1977). Color Edge Detection. *Optimal Engineering* , 479-484.
- Shilland, G. R. (2009). *Host-Based Multivariate Statistical Computer Operating Process Anomaly Intrusion Detection System (PAIDS)*. Wright Patterson Air Force Base OH: MS Thesis, AFIT/GOR/ENS/09-15. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).
- Smetek, T. E. (2007). *Hyperspectral Imagery Target Detection Using Improved Anomaly Detection and Signature Matching Methods*. Wright Patterson Air Force Base, Ohio: Air Force Institute of Technology.
- Smetek, T. E., & Bauer, K. W. (2008). A Comparison of Multivariate Outlier Detection Methods for Finding Hyperspectral Anomalies. *Military Operations Research* , 19-43.
- Smetek, T. E., & Bauer, K. W. (2007). Finding Hyperspectral Anomalies Using Multivariate Outlier Detection. *Aerospace Conference* (pp. 1-24). Big Sky, Montana: IEEE.
- Taitano, Y. P., & Bauer, K. W. (2010). A Locally Adaptable Iterative RX Detector. *EURASIM Journal on Advanced Signal Processing, Special Issue on Advanced Image Processing for Defense and Security Applications* , 1-10.
- Tang, J., Yu, W., Chai, T., & Zhao, L. (2012). On-line Principal Component Analysis with Application to Process Modeling. *Neurocomputing* , 167-178.
- Tarabalka, Y., Benediktsson, J. A., & Chanussot, J. (2009). Spectral-Spatial Classification of Hyperspectral Imagery Based on Partitional Clustering Techniques. *IEEE Transaction on Geoscience and Remote Sensing* , 2973-2987.
- Trucco, E., & Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, New Jersey: Prentice-Hall.
- Turnquist, B. R. (2011). *Fusion Schemes for Ensembles of Hyperspectral Anomaly Detection Algorithms*. Wright Patterson Air Force Base OH: MS Thesis, AFIT-OR-MS-ENS-11-25. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).

- Waddell, E. T. (2003). *An analysis of camera calibration for voxel coloring including the effect of calibration on voxelization errors*. Lexington: MS Thesis. University of Kentucky.
- Wang, W., Masegla, F., Guyet, T., Quiniou, R., & Cordier. (2009). A General Framework for Adaptive and Online Detection of Web Attacks. *WWW 2009 Proceedings* (pp. 1141-1142). Madrid, Spain: ACM.
- Williams, J. P. (2007). *Robustness of Multiple Clustering Algorithms on Hyperspectral Images*. Wright Patterson Air Force Base: MS thesis, AFIT/GOR/ENS/07-27. Graduate School of Engineering and Management, Air Force Institute of Technology (AU).
- Williams, J. P. (2012). *Towards the Mitigation of Correlation Effects in the Analysis of Hyperspectral Imagery with Extensions to Robust Parameter Design*. Wright Patterson Air Force Base, Ohio: Air Force Institute of Technology.
- Woodruff, D. L., & Reiners, T. (2004). Experiments With, and On, Algorithms for Maximum Likelihood Clustering. *Computational Statistics and Data Analysis* , 237-253.
- Xu, R., & Wunsch, D. (2005). Survey of Clustering Algorithms. *IEEE Transactions on Neural Networks* , 645-678.

Vita

Maj Waddell graduated as the Salutatorian of his class at Boyd County High School in Ashland, Kentucky before attending the University of Kentucky, graduating *summa cum laud* with Bachelor of Science Degrees in Computer Science and Mathematics. He later also completed a Master of Science degree in Computer Science. Following the conclusion of his course work, he was commissioned as a Second Lieutenant in the United States Air Force as a distinguished graduate of the Reserve Officer Training Corps program. After his initial assignment as an electronic warfare Operations Analyst, he attended the USAF Test Pilot School (TPS) as a Flight Test Engineer (FTE), graduating as the Onizuka Prop Wash award winner. Maj Waddell then served as an FTE on numerous flight test programs, flying over 500 hours in over 30 aircraft including the B-52, C-12, F-16, T-38, KC-135, and C-130. He also served as a USAF TPS instructor. His assignment immediately prior to attending AFIT was as the Director of Operations and Deputy Branch Chief for the Space Test and Operations Branch of the Space and Missile Systems Center at Kirtland AFB, New Mexico. He was responsible for the acquisition, maintenance, operation, and world-wide deployment of unique space telemetry systems supporting numerous programs. While at Kirtland, Maj Waddell deployed in support of Operations IRAQI FREEDOM and NEW DAWN. He was selected as the 2010 recipient of the Wayne P. Hughes award by the Military Operations Research Society. He is also a graduate of the Air and Space Basic Course, Squadron Officer School, and Air Command and Staff College (correspondence program). Upon graduation, Maj Waddell will be assigned as a Program Element Monitor on the staff of the Assistant Secretary of the Air Force for Acquisition.

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 074-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>					
1. REPORT DATE (DD-MM-YYYY) 13-06-2013		2. REPORT TYPE Graduate Research Paper		3. DATES COVERED (From – To) May 2012 – 13 June 2013	
4. TITLE AND SUBTITLE Online Cluster Analysis Supporting Real Time Anomaly Detection in Hyperspectral Imagery				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Waddell, Elwood T. Jr., Major, USAF				5d. PROJECT NUMBER N/A	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENS-GRP-13-J-25	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RY ATTN: Lt Col David Ryer 2241 Avionic Circle, Area B WPAFB OH 45433 937-938-8389 (DSN 798)				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RY	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT Ongoing work in anomaly detection in hyperspectral images has shown that cluster analysis performed in appropriate principal component subspaces can enhance the performance of detectors such as the Reed-Xiaoli detector and its derivatives. Numerous operational considerations motivate the development of an online or incremental clustering algorithm, which can perform clustering as pixels of the image are collected in real time rather than waiting until the full image is complete. Such an algorithm is developed by combining key elements of existing clustering algorithms from related domains. The parameters of the algorithm are tuned and performance of the algorithm is assessed using a set of actual hyperspectral images by exploiting key attributes of an appropriate principal component sub-space. A byproduct of the clustering algorithm is a rudimentary anomaly detector which demonstrates the feasibility of cluster based outlier detection in hyperspectral imagery.					
15. SUBJECT TERMS Hyperspectral Imagery, Online Cluster Analysis, Principal Component Analysis, Anomaly Detection					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 172	19a. NAME OF RESPONSIBLE PERSON Bauer, Kenneth, Ph.D.
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) (937) 255-3636, x 4943 (Kenneth.Bauer@afit.edu)